

SIEMENS

在 VC 中如何实现 OPC 数据访问

How to achieve data access through OPC in VC

Getting-started

Edition (2009 年 06 月)

摘要 本文主要讲述了在 VC 语言环境下，编程实现通过 SimaticNet 提供的 OPC Server，访问 PLC 中数据的步骤，此方法同样适用于 WinCC 作为 OPC Server 时的数据访问。

关键词 SimaticNet、VC、OPC、WinCC

Key Words SimaticNet、VC、OPC、WinCC

在VC中如何实现OPC数据访问	1
1、概述	4
1.1 OPC介绍	4
1.2 OPC的读写方式	5
1.3 OPC访问接口方式	6
2、测试环境	7
2.1 硬件要求	7
2.2 软件要求	7
3、OPC Server端组态配置	7
4、用VC自定义设计过程	9
4.1 同步读写	9
4.2 异步读写	16
5、OPCItem的数据类型	24
6、小结	24
7、代码	24
7.1 异步读写（包括订阅）	24

1、概述

1.1 OPC 介绍

OPC 是 Object Linking and Embedding (OLE) for Process Control 的缩写，它是微软公司的对象链接和嵌入技术在过程控制方面的应用。OPC 以 OLE/COM/DCOM 技术为基础，采用客户/服务器模式，为工业自动化软件面向对象的开发提供了统一的标准，这个标准定义了应用 Microsoft 操作系统在基于 PC 的客户机之间交换自动化实时数据的方法，采用这项标准后，硬件开发商将取代软件开发商为自己的硬件产品开发统一的 OPC 接口程序，而软件开发者可免除开发驱动程序的工作，充分发挥自己的特长，把更多的精力投入到其核心产品的开发上。

SimaticNet 是西门子全集成自动化系统中的一个重要组成部分，它为完善的工业自动化控制系统的通讯提供部件和网络，同时提供多个 OPCServer，为数据的外部访问提供接口，本文主要以 OPC.SimaticNET 为例说明。

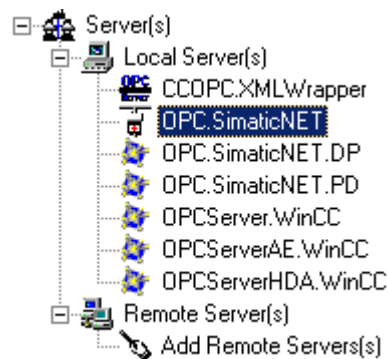


图 1: SimaticNet 提供的 OPCServer

采用不同的通信方式，通过 OPC.SimaticNET，现场数据可以方便地提供给用户：

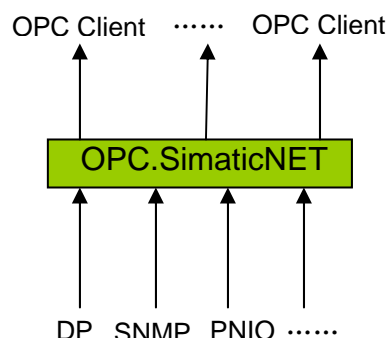


图 2: 多种数据提供方式

1.2 OPC 的读写方式

在实际使用中，主要包括对现场数据的读写操作。

OPC 客户端读取数据有三种方式：同步、异步、订阅。

同步通讯时，OPC 客户程序向 OPC 服务器进行请求时，OPC 客户程序必须等到 OPC 服务器对应的响应全部完成以后才能返回，在此期间 OPC 客户程序一直处于等待状态，若进行读操作，那么必须等待 OPC 服务器响应后才返回。因此在同步通讯时，如果有大量数据进行操作或者有很多 OPC 客户程序对 OPC 服务器进行读操作，必然造成 OPC 客户程序的阻塞现象。因此同步通讯适用于 OPC 客户程序较少，数据量较小时的场合。

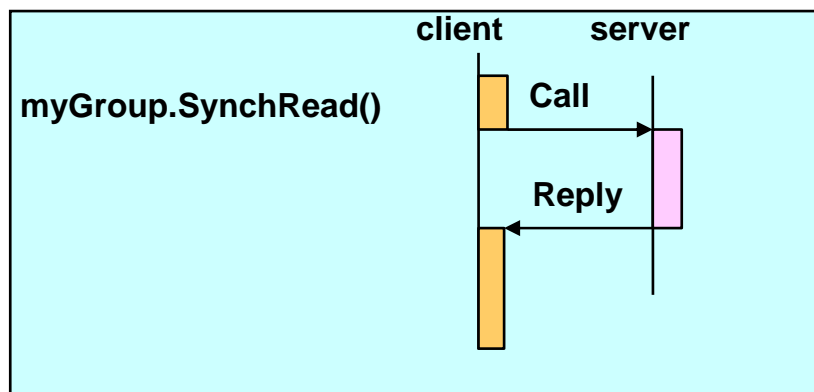


图 3 OPC 同步读写服务器-客户端数据流图

异步通讯时，OPC 客户程序对服务器进行请求时，OPC 客户程序请求后立刻返回，不用等待 OPC 服务器的响应，可以进行其它操作。OPC 服务器完成响应后再通知 OPC 客户程序，如进行读操作，OPC 客户程序通知 OPC 服务器后离开返回，不等待 OPC 服务器的读完成，而 OPC 服务器完成读后，会自动的通知 OPC 客户程序，把读结果传送给 OPC 客户程序。因此相对于同步通讯，异步通讯的效率更高。

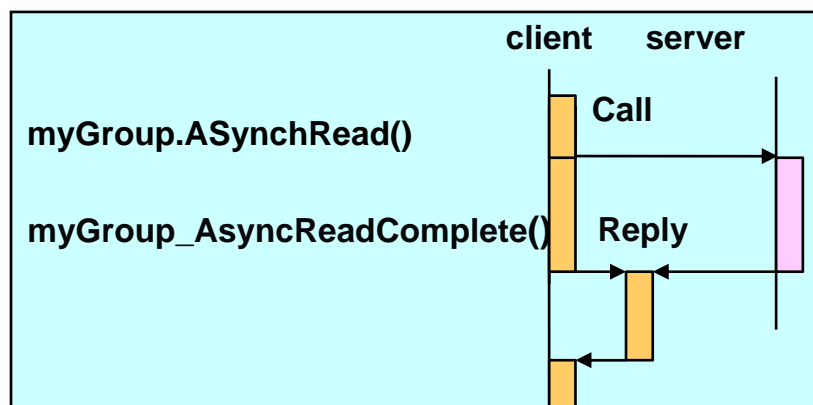


图 4 OPC 异步读服务器-客户端数据流图

订阅方式时，OPC 客户程序对服务器进行请求时，OPC 客户程序操作后立刻返回，不用等待 OPC 服务器的操作，可以进行其它操作， OPC 服务器的 Group 组在组内有数据发生改变时，自动根据更新周期刷新相应的客户端数据，如下图，客户端只向 OPC 服务发送一次请求，之后不再对服务器请求。

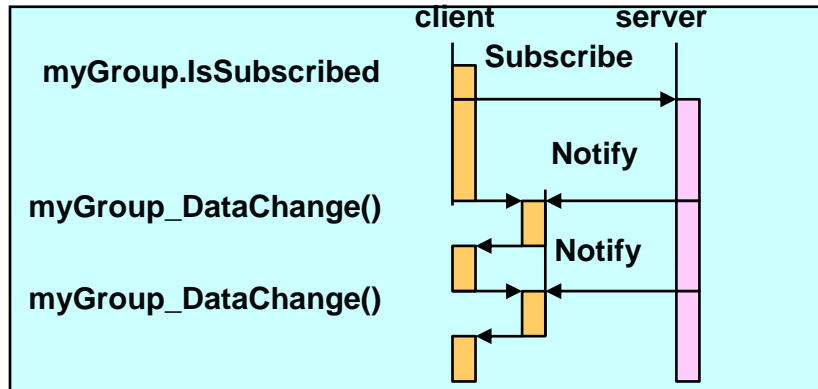


图 5 OPC 同步读服务器-客户端数据流图

OPC 写数有两种方式：同步、异步。区别与上面讲的机制一样，在生产应用中，如果写数据参与控制，一般采用同步方式。

1.3 OPC 访问接口方式

OPC 主要包含两种接口：CUSTOM 标准接口和 OLE 自动化标准接口，自定义接口是服务商必须提供的，而自动化接口则是可选的。

自定义接口是一组 COM 接口，主要用于采用 C++语言的应用程序开发；

自动化接口是一组 OLE 接口，主要用于采用 VB，DELPHI，Excel 等基于脚本编程语言的应用程序开发。

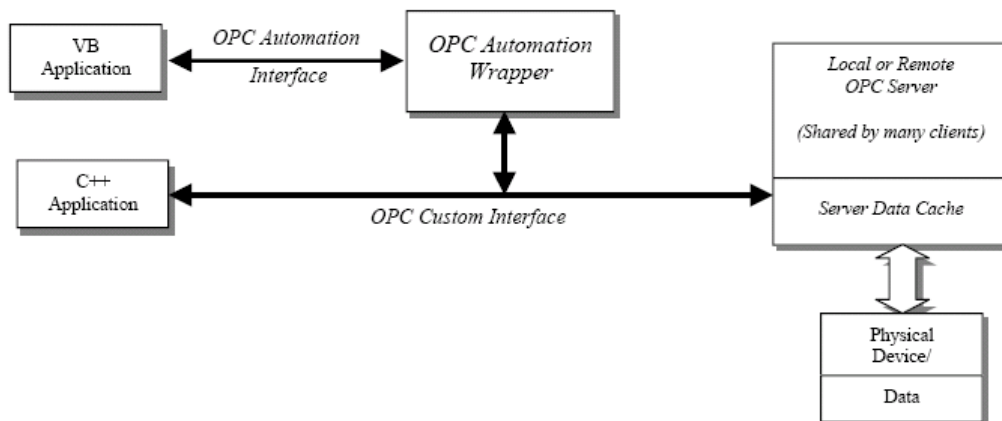


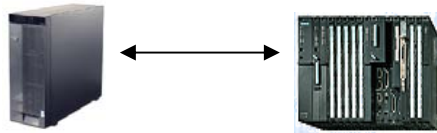
图 6 自定义接口和自动化接口

本文中通过实验，基于自定义接口，逐步讲解了通过 VC 编写客户端程序，访问 OPC.SimaticNet，对 PLC 数据进行读写的实现过程。

2、测试环境

2.1 硬件要求

采用 400 系列 PLC，通过以太网连接到安装有 simaticNet 的计算机上。



computer: windows 2003 server-----192.168.0.102

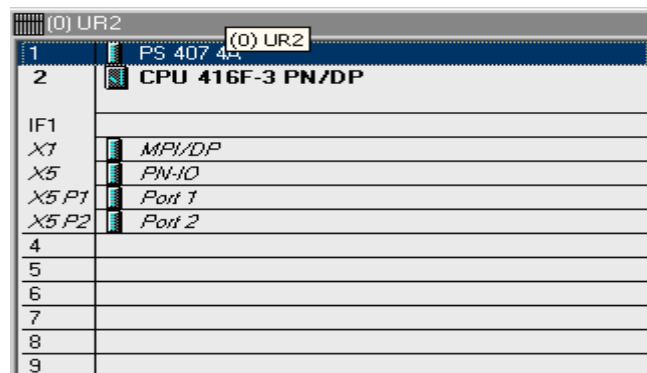
CPU: CPU414-3PN -----416-3FR05-0AB0-----192.168.0.1

2.2 软件要求

computer:

- ✓ Simatic.net 2007
- ✓ Visual studio 2005
- ✓ Step7 V5.4 SP4

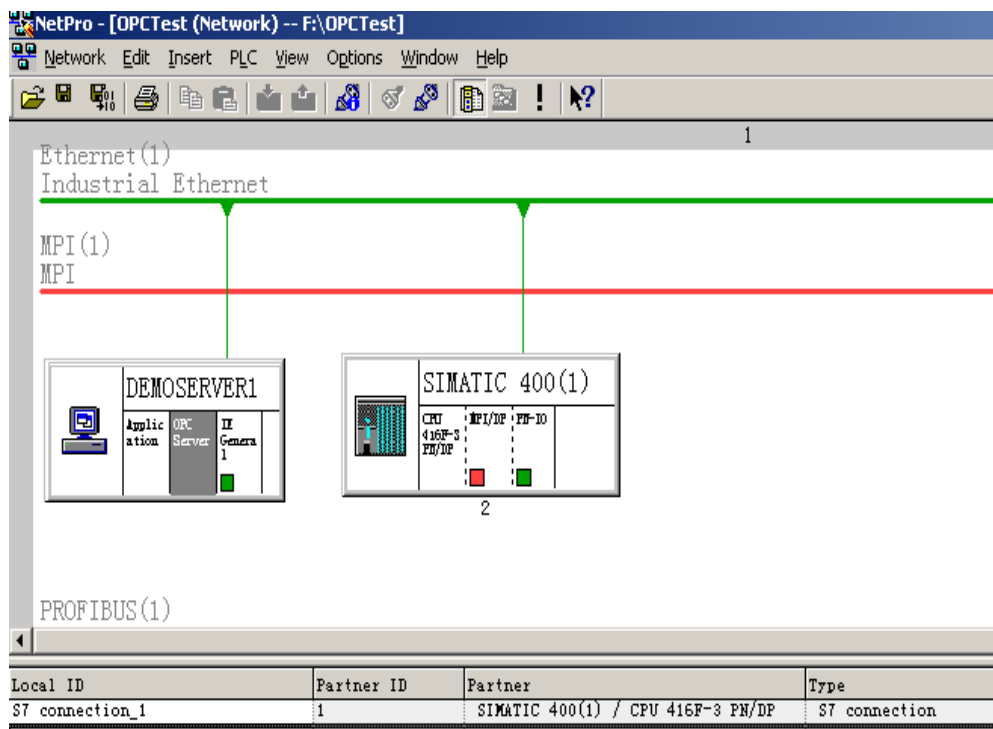
3、OPC Server 端组态配置



在 CPU 中定义 DB 块: DB10

Address	Name	Type	Initial value
0.0		STRUCT	
+0.0	Test_Data3	INT	0
+2.0	Test_Data4	INT	0
+4.0	Test_Data5	REAL	0.000000e+000
+8.0	Test_Data6	REAL	0.000000e+000
+12.0	Test_Data7	BOOL	FALSE
+12.1	Test_Data8	BOOL	FALSE
+14.0	Test_Data9	STRING[10]	''
+26.0	Test_Data10	STRING[10]	''
=38.0		END_STRUCT	

配置 PC Station，参考其它文档。



如上图建立连接 S7_connection_1，然后在 OPC Scout 测试连接的正确性。

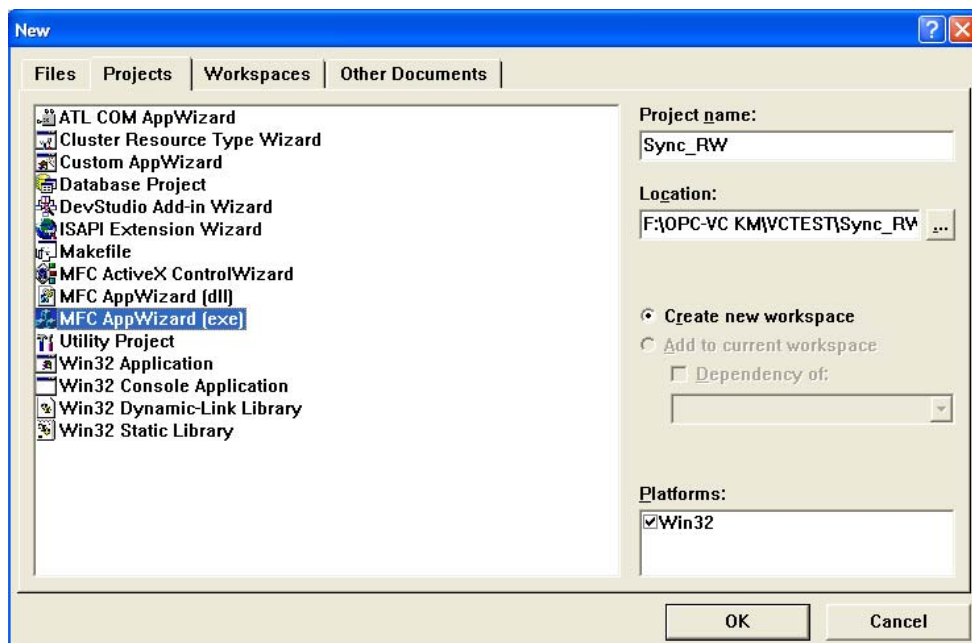
	Item Names	Value	Format	Type	Access	Quality	Stamp (t
1	S7:[S7 connection_1]DB10,INT0	2	Original	int16	RW	good	03/18/2009
2	S7:[S7 connection_1]DB10,INT2	4	Original	int16	RW	good	03/18/2009
3	S7:[S7 connection_1]DB10,REAL4	3.5	Original	real32	RW	good	03/18/2009
4	S7:[S7 connection_1]DB10,REAL8	5.8	Original	real32	RW	good	03/18/2009
5	S7:[S7 connection_1]DB10,STRING14.10	test	Original	string	RW	good	03/18/2009
6	S7:[S7 connection_1]DB10,STRING26.10	20081213	Original	string	RW	good	03/18/2009
7	S7:[S7 connection_1]DB10,X12.0	True	Original	bool	RW	good	03/18/2009
8	S7:[S7 connection_1]DB10,X12.1	False	Original	bool	RW	good	03/18/2009

从上面可以看到数据访问都是正常的。

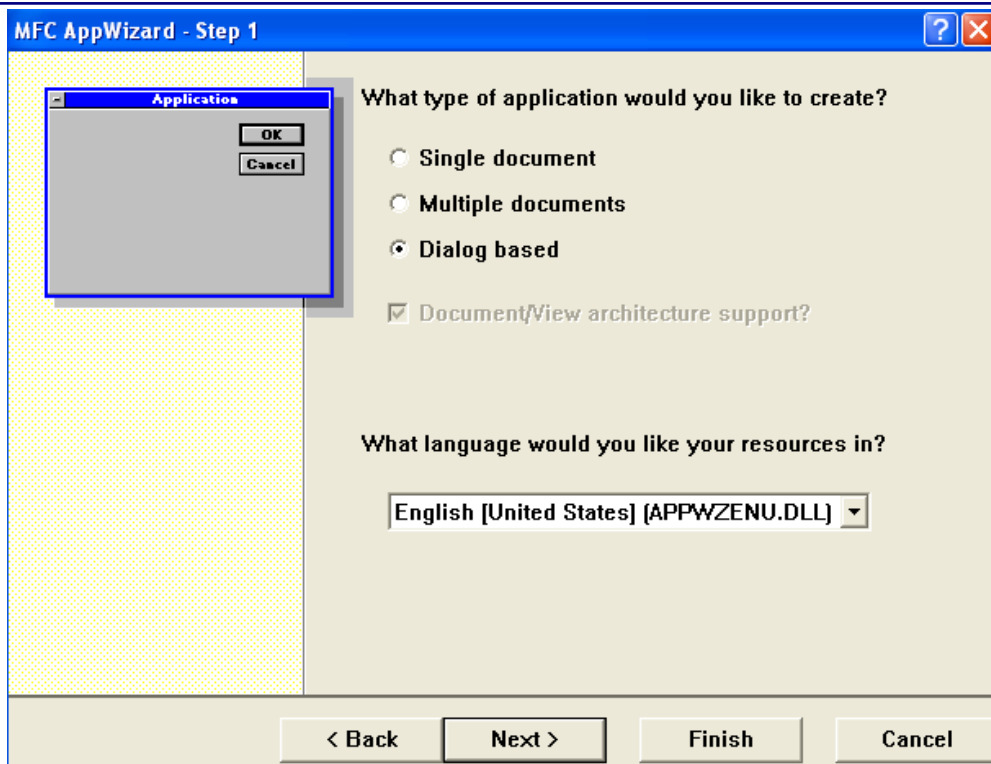
4、用 VC 自定义设计过程

4.1 同步读写

建立项目：命名为 Sync_RW



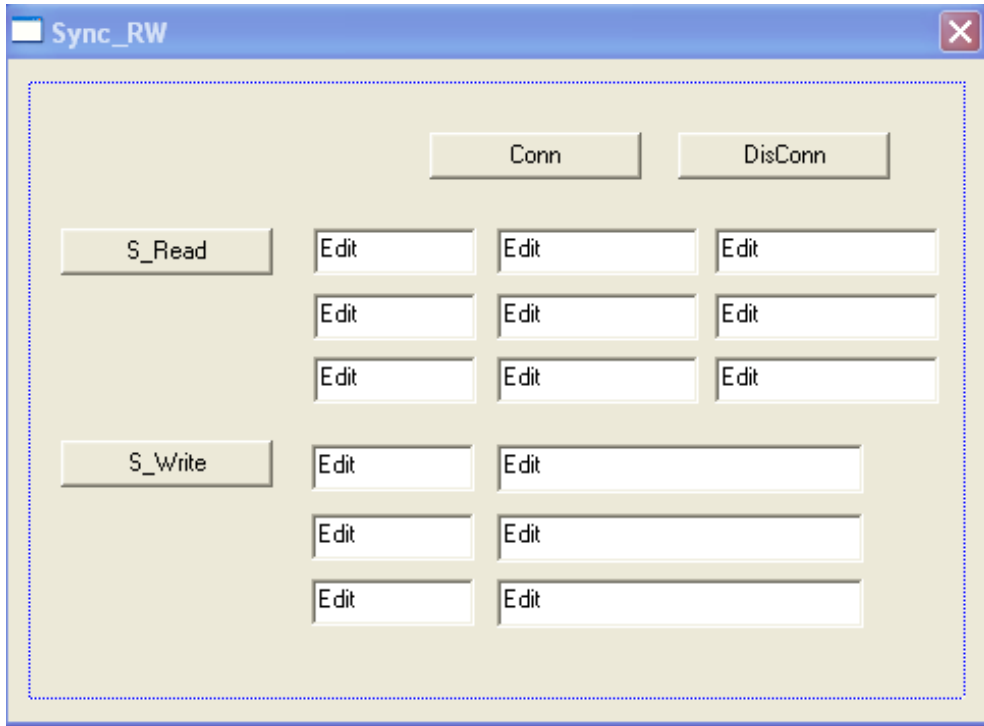
如下，建立工程项目为 Dialog based 项目类型。



如下：在 IDD_OPEDA_SYNC_DIALOG 对话框中作如下定义：

Control	name	Text
Button:	IDC_BUTTON_Conn	Conn
Button:	IDC_BUTTON_SRead	S_Read
Button:	IDC_BUTTON_SWrite	S_Write
Button:	IDC_BUTTON_DisConn	DisConn
EditBox:	IDC_EDIT_ReadVal1	
EditBox:	IDC_EDIT_ReadVal2	
EditBox:	IDC_EDIT_ReadVal3	
EditBox:	IDC_EDIT_ReadQu1	
EditBox:	IDC_EDIT_ReadQu2	
EditBox:	IDC_EDIT_ReadQu3	
EditBox:	IDC_EDIT_ReadTS1	
EditBox:	IDC_EDIT_ReadTS2	
EditBox:	IDC_EDIT_ReadTS3	
EditBox:	IDC_EDIT_WriteVal1	
EditBox:	IDC_EDIT_WriteVal2	
EditBox:	IDC_EDIT_WriteVal3	

TextBox: IDC_EDIT_WriteRes1
 TextBox: IDC_EDIT_WriteRes2
 TextBox: IDC_EDIT_WriteRes3



第一步，在 CSync_RWDlg 中增加相应变量定义：



```

CString m_WriteRes3;
CString m_WriteRes2;
CString m_WriteRes1;
CString m_WriteVal3;
double m_WriteVal2;
  
```

```

int m_WriteVal1;
CString m_ReadTs3;
CString m_ReadTs2;
CString m_ReadTs1;
CString m_ReadQu3;
CString m_ReadQu2;
CString m_ReadQu1;
CString m_ReadVal3;
double m_ReadVal2;
int m_ReadVal1;
HICON m_hIcon;

```

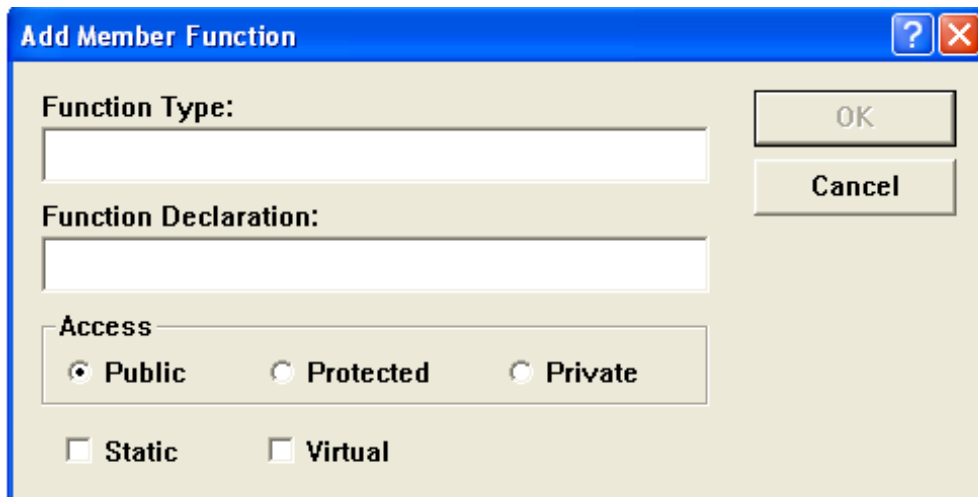
同时增加 OPC 自定义标准中接口变量

```

IOPCServer      *m_pIOPCServer; // OPCServer 句柄
IOPCItemMgt     *m_pIOPCItemMgt; //OPCItem 状态句柄
IOPCSyncIO      *m_pIOPCSyncIO;
OPCITEMDEF      m_Items[3];
OPCITEMRESULT   *m_pItemResult;
OPCHANDLE       m_GrpSrvHandle;
HRESULT          *m_pErrors;

```

第二步，在 CSync_RWDlg 中增加相应函数定义：



```

afx_msg void OnConn();
afx_msg void OnWrite();
afx_msg void OnRead();
afx_msg void OnDisConn();

```

第三步，将变量及动作在 DoDataExchange, BEGIN_MESSAGE_MAP 建立联系：

```
void void CSync_RWDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT_ReadVal1, m_ReadVal1);
    .....//增加相应代码
}
BEGIN_MESSAGE_MAP(CSync_RWDlg, CDialog)
    ON_BN_CLICKED(IDC_BUTTON_Conn, OnConn)
    .....//增加相应代码
END_MESSAGE_MAP()
```

第四步，增加 OPC 相应的外部引用文件：

增加文件： Pre_OPC.h ， Pre_OPC.cpp

在 Pre_OPC.h 中，增加 OPC 的通用头文件：

```
#include "OPCDA.H"
#include "OPCcomn.H"
#include "OPCError.h"
```

注意在 Sync_RW.cpp 及 Sync_RWDlg.cpp 中引用"Pre_OPC.h"时，按照如下顺序：

```
#include "stdafx.h"
#include "Pre_OPC.h"
#include "Sync_RW.h"
#include "Sync_RWDlg.h"
```

第五步，初始化 Com 控件，注册 OPCGroup 及 OPCItem：

```
void CSync_RWDlg::OnConn()
{
    .....
    HRESULT          r1;
    r1 = CLSIDFromProgID(L"OPC.SimaticNET", &clsid);
    r1 = CoCreateInstance (clsid, NULL, CLSCTX_LOCAL_SERVER ,
                          IID_IOPCServer, (void**)&m_pIOPCServer);
    // "OPC.SimaticNet", "192.168.0.102"是OPCServer 名称及所在 computer 地址
    // CoCreateInstance 创建一个 OPCServer 的实例
```

```

.....
r1=m_pIOPCServer->AddGroup()//增加相应的组，定义组的特性，并输出组的句柄
.....
m_Items[0].szAccessPath    = L"";
m_Items[0].szItemID       = L"S7:[@LOCALSERVER]DB1,INT0";
                           //地址，不同数据类型表示方法不同
m_Items[0].bActive        = TRUE; //是否激活
m_Items[0].hClient        = 1; //标示 ID，不同的 Item 不一样
m_Items[0].dwBlobSize     = 0;
m_Items[0].pBlob          = NULL;
m_Items[0].vtRequestedDataType = 2; //数据类型表示格式，2 表示 int
.....
r1 = m_pIOPCItemMgt->AddItems(3,m_Items,&m_pItemResult,
                             &m_pErrors); //将定义的 OPCItem 加入组内，注意数量
.....
r1 = m_pIOPCItemMgt->QueryInterface(IID_IOPCSyncIO,
                                   (void**)&m_pIOPCSyncIO);
                           //为 OPCGroup 的组对象定义接口
.....
}

```

第六步，同步读：

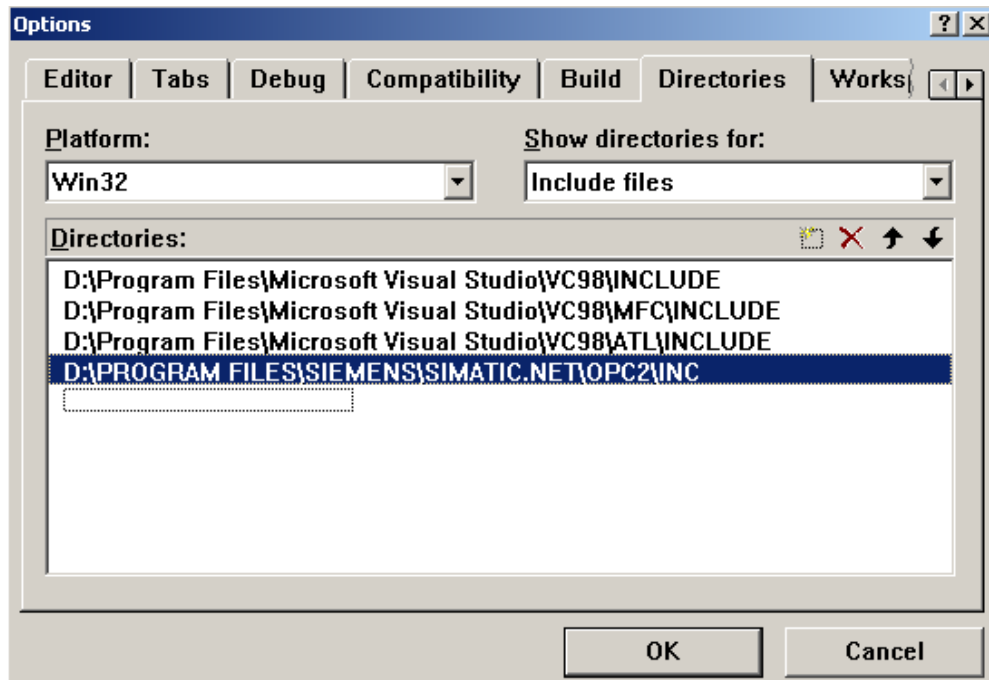
```

void CSync_RWDlg::OnRead()
{
.....
r1 = m_pIOPCSyncIO->Read(OPC_DS_DEVICE, 3, phServer,
                        &pItemValue, &pErrors); //同步读
m_ReadVal1 = pItemValue[0].vDataValue.lVal; //得出值
qnr = pItemValue[0].wQuality; //
m_ReadQual = GetQualityText(qnr); //质量码
m_ReadTsl = COleDateTime( pItemValue[0].ftTimeStamp ).Format();
                           //时间
UpdateData(FALSE); //更新数据到画面
.....
}

```

注意，编译程序，出现错误：

LINK : fatal error LNK1168: cannot open Debug/Sync_RW.exe for writing, 是由于工程的路径不对，需要工具->选择->目录中，如下添加即可：



在测试中，采用了三种数据类型，分别为 int, real, string，在程序设计时要注意。具体参建后面说明。

第七步，同步写：

```
void CSync_RWDlg::OnWrite()
{
    .....
    values[0].vt = VT_I2;
    values[0].iVal = m_WriteVal1; //得出值
    r1 = m_pIOPCSyncIO->Write(3, phServer, values, &pErrors); //同步写
    .....
    UpdateData(FALSE);
    .....
}

```

第八步，释放对象：

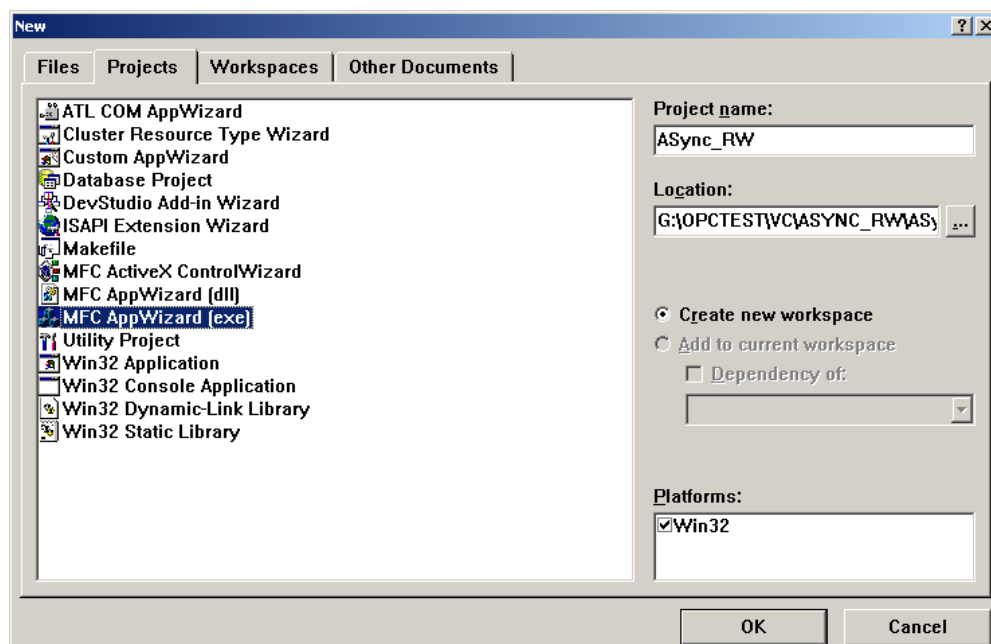
```
void CSync_RWDlg::OnDisConn()
{
    .....
}

```

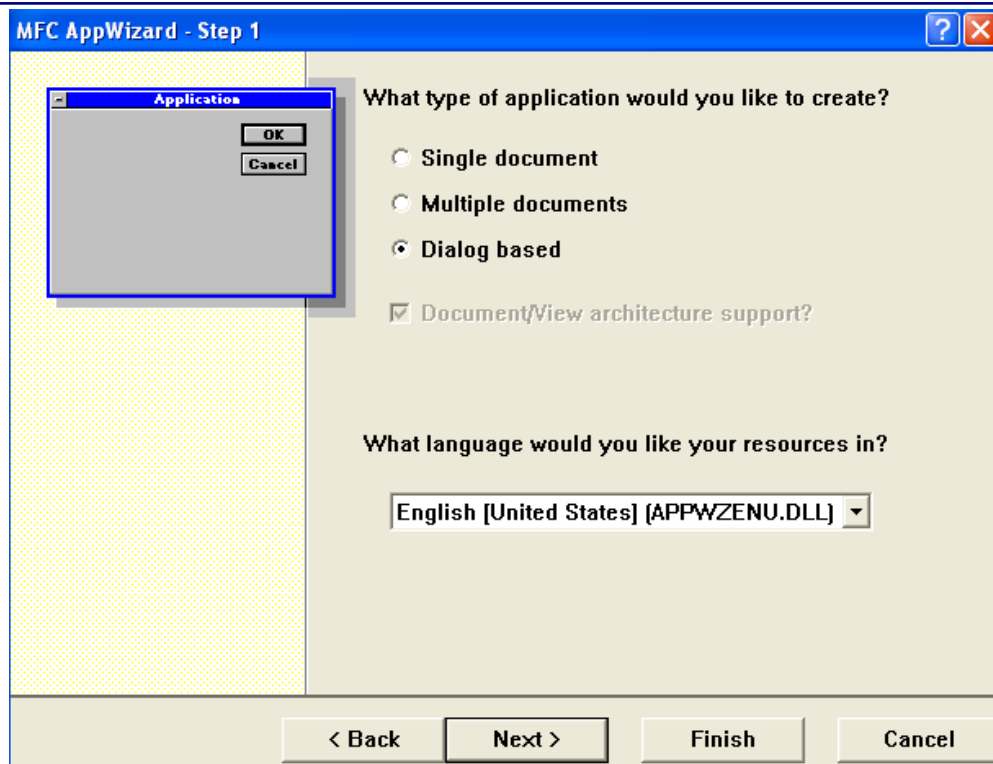
```
r1 = m_pOPCItemMgt->RemoveItems(3,phServer,&pErrors);  
                                     //删除 OPCItem  
.....  
r1=m_pOPCServer->RemoveGroup(m_GrpSrvHandle, TRUE);  
.....                                     //删除 OPCGroup  
m_pOPCServer->Release();                 //释放 OPCServer  
.....  
}
```

4.2 异步读写

建立项目：命名为 ASync_RW



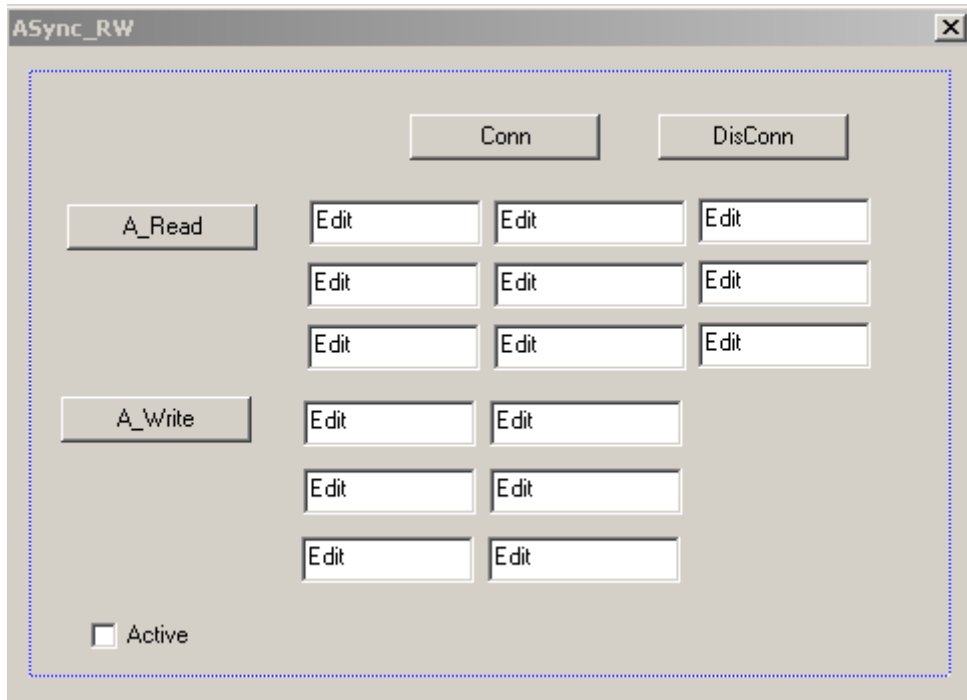
如下，建立工程项目为 Dialog based 项目类型。



如下：在 IDD_OPEDA_SYNC_DIALOG 对话框中作如下定义：

Control	name	Text
Button:	IDC_BUTTON_Conn	Conn
Button:	IDC_BUTTON_ARead	A_Read
Button:	IDC_BUTTON_AWrite	A_Write
Button:	IDC_BUTTON_DisConn	DisConn
EditBox:	IDC_EDIT_ReadVal1	
EditBox:	IDC_EDIT_ReadVal2	
EditBox:	IDC_EDIT_ReadVal3	
EditBox:	IDC_EDIT_ReadQu1	
EditBox:	IDC_EDIT_ReadQu2	
EditBox:	IDC_EDIT_ReadQu3	
EditBox:	IDC_EDIT_ReadTS1	
EditBox:	IDC_EDIT_ReadTS2	
EditBox:	IDC_EDIT_ReadTS3	
EditBox:	IDC_EDIT_WriteVal1	
EditBox:	IDC_EDIT_WriteVal2	
EditBox:	IDC_EDIT_WriteVal3	

EditBox: IDC_EDIT_WriteRes1
 EditBox: IDC_EDIT_WriteRes2
 EditBox: IDC_EDIT_WriteRes3
 CheckBox: IDC_CHK_Active



第一步，在 CASync_RWDlg 中增加相应变量定义：



```

CString m_WriteRes3;
CString m_WriteRes2;
CString m_WriteRes1;
CString m_WriteVal3;
double m_WriteVal2;
  
```

```

int m_WriteVal1;
CString m_ReadTs3;
CString m_ReadTs2;
CString m_ReadTs1;
CString m_ReadQu3;
CString m_ReadQu2;
CString m_ReadQu1;
CString m_ReadVal3;
double m_ReadVal2;
int m_ReadVal1;
BOOL m_ActiveCheck;

```

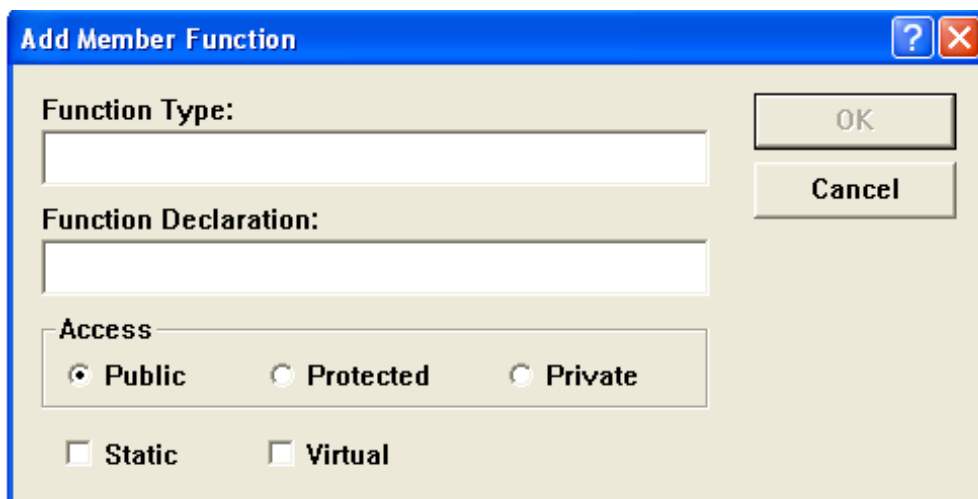
同时增加 OPC 自定义标准中接口变量

```

DWORD                m_dwAdvise;
OPCITEMDEF           m_Items[3];
IOPCServer           *m_pIOPCServer; // OPCServer 句柄
IOPCItemMgt          *m_pIOPCItemMgt; //OPCItem 状态句柄
IOPCGroupStateMgt   *m_pIOPCGroupStateMgt;
IOPCAsyncIO2         *m_pIOPCAsyncIO2;
OPCITEMRESULT        *m_pItemResult;
HRESULT              *m_pErrors;
OPCHANDLE             m_GrpSrvHandle;

```

第二步，在 CASync_RWDlg 中增加相应函数定义：



```

afx_msg void OnConn();
afx_msg void OnWrite();

```

```
afx_msg void OnRead();
afx_msg void OnCHECKData();
afx_msg void OnDisConn();
```

第三步，将变量及动作在 DoDataExchange, BEGIN_MESSAGE_MAP 建立联系：

```
void void CASync_RWDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT_ReadVal1, m_ReadVal1);
    .....//增加相应代码
}
BEGIN_MESSAGE_MAP(CASync_RWDlg, CDialog)
    ON_BN_CLICKED(IDC_BUTTON_Conn, OnConn)
    .....//增加相应代码
END_MESSAGE_MAP()
```

第四步，增加 OPC 相应的外部引用文件：

增加文件： Pre_OPC.h ， Pre_OPC.cpp

增加文件： Callback.h, Callback.cpp

在 Pre_OPC.h 中，增加 OPC 的通用头文件：

```
#include "OPCDA.H"
#include "OPCcomn.H"
#include "OPCError.h"
```

注意在 ASync_RW.cpp 及 ASync_RWDlg.cpp 中引用"Pre_OPC.h"时，按照

如下顺序：

```
#include "stdafx.h"
#include "Pre_OPC.h"
#include "ASync_RW.h"
#include "ASync_RWDlg.h"
```

编译如果报错 fatal error C1010: unexpected end of file while looking for
precompiled header directive

则在 callback.cpp 前面加编译文件#include "stdafx.h"。

编译如果 error LNK2001: ATL::CComModule _Module"
(?_Module@ @3VCComModule@ATL@ @A)

则在"ASync_RW.h"前面增加 CComModule _Module;

第五步，初始化 Com 控件，注册 OPCGroup 及 OPCItem:

```

void CASync_RWDlg::OnConn()
{
    .....
    HRESULT          r1;
    r1 = CLSIDFromProgID(L"OPC.SimaticNET", &clsid);
    r1 = CoCreateInstance (clsid, NULL, CLSCTX_LOCAL_SERVER ,
                          IID_IOPCServer, (void**)&m_pIOPCServer);
    // "OPC.SimaticNet", "192.168.0.102"是 OPCServer 名称及所在 computer 地址
    // CoCreateInstance 创建一个 OPCServer 的实例
    .....
    r1=m_pIOPCServer->AddGroup()//增加相应的组，定义组的特性，并输出组的句柄
    .....
    m_Items[0].szAccessPath      = L"";
    m_Items[0].szItemID         = L"S7:[@LOCALSERVER]DB1,INT0";
    //地址，不同数据类型表示方法不同
    m_Items[0].bActive          = TRUE; //是否激活
    m_Items[0].hClient          = 1; //标示 ID，不同的 Item 不一样
    m_Items[0].dwBlobSize       = 0;
    m_Items[0].pBlob            = NULL;
    m_Items[0].vtRequestedDataType = 2; //数据类型表示格式，2 表示 int
    .....
    r1 = m_pOPCItemMgt->AddItems(3,m_Items,&m_pltemResult,
                                &m_pErrors);//将定义的 OPCItem 加入组内，注意数量
    .....
    r1 = m_pOPCItemMgt->QueryInterface(IID_IOPCSyncIO,
                                       (void**)&m_pIOPCSyncIO);
    //为 OPCGroup 的组对象定义接口
    .....
    r1 = m_pOPCItemMgt->QueryInterface(IID_IOPCAsyncIO2,
                                       (void**)&m_pIOPCAsyncIO2);
    //为 OPCGroup 的异步读对象定义接口
    .....
    CComObject<COPCDataCallback>* pCOPCDataCallback; //Callback 事件指针
    .....
}

```

第六步，异步读：

```
void CASync_RWDlg::OnRead()
{
    .....
    r1 = m_pIOPCAsyncIO2->Read(3,phServer,10, &dwCancelID, &pErrors);
        //异步读
    .....
}
```

异步读操作后，通过事件回调函数 OnReadComplete，得到相应数据

```
STDMETHODIMP COPCDataCallback::OnReadComplete ()
{
    .....
    if (pErrors[0] == S_OK)
    {
        .....
        m_pCDlgClass->m_ReadVal1 =pvValues[0].iVal;    //数值
        m_pCDlgClass->m_ReadQu1 =readQuality;    //质量码
        m_pCDlgClass->m_ReadTs1 =readTS;    //时间
        .....
    }
    m_pCDlgClass->UpdateData(FALSE); //
}
```

第七步，订阅读：

```
void CASync_RWDlg::OnCHECKData()
{
    HRESULT r1= m_pIOPCGroupStateMgt->SetState(,
        &dwRevUpdateRate, //更新速度
        &m_ActiveCheck, //订阅激活
        ...)
}
```

订阅读操作后，通过事件回调函数 OnReadComplete，得到相应数据

```
STDMETHODIMP COPCDataCallback:: OnDataChange()
{
    for (i = 0; i<dwCount; i++)
```

```

{
    switch (phClientItems[i])
    {
        case 1:
            m_pCDlgClass->m_ReadVal1 =pvValues[i].iVal; //数值
            m_pCDlgClass->m_ReadQ=GetQualityText(pwQualities[0]); //质量码
            m_pCDlgClass->m_ReadTs1=COleDateTime( pftTimeStamps[0] ).Format();
            break;
            .....
        }
    }
}

```

在这里要注意，对于 phClientItems[i]是 Item 的 ID 号。一定要对应上。

第八步，异步写：

```

void CASync_RWDlg::OnWrite()
{
    .....

    r1 = m_pIOPCAsyncIO2->Write(2,phServer,values,2,&dwCancelID,&pErrors)
                                           //异步写

    .....
}

```

写操作后，通过事件回调函数 OnWriteComplete，得到处理结果

STDMETHODIMP COPCDataCallback::OnWriteComplete ()

第九步，释放对象：

```

void CASync_RWDlg::OnDisConn()
{
    .....

    r1 = m_pIOPCItemMgt->RemoveItems(3,phServer,&pErrors);
                                           //删除 OPCItem

    .....

    r1=m_pIOPCServer->RemoveGroup(m_GrpSrvHandle, TRUE);
    ..... //删除 OPCGroup
    m_pIOPCServer->Release(); //释放 OPCServer
    .....
}

```

5、OPCItem 的数据类型

```

m_Items[0].szAccessPath    = L"";
m_Items[0].szItemID       = L"S7:[@LOCALSERVER]DB1,INT0";
                               //地址，不同数据类型表示方法不同
m_Items[0].bActive        = TRUE; //是否激活
m_Items[0].hClient        = 1; //标示 ID，不同的 Item 不一样
m_Items[0].dwBlobSize    = 0;
m_Items[0].pBlob          = NULL;
m_Items[0].vtRequestedDataType = 2; //数据类型表示格式，2 表示 int

```

在上面可以看到，vtRequestedDataType 代表了不同数据类型，在使用中需要注意的。

在取数据时对于 vDataValue 结构分析，各种数据类型对应关系如下：

VbBoolean	VbByte	VbDecimal	VbDouble	Vbinteger	VbLong	VbSingle	VbString
11	17	14	5	2	3	4	8
boolVal	bVal		dblVal	iVal	lVal	fltVal	bstrVal

6、小结

在实际应用中，根据实际要求，合理选择读写方式是很重要的。同时实例中是以 SimaticNet 的 OPCServer 为例，对于 WinCC 作为 OPCServer 同样适用，只需要将 "OPC.SimaticNet" 改为 "OPCServer.WinCC"。

7、代码

下面是异步读写（包括订阅）的代码，不包含自动生成文件，同步方式读写可参考。在西门子 Simatic 安装盘中也有类似例程可参考。

7.1 异步读写（包括订阅）

```

//Pre_OPC.h
#define VC_EXTRALEAN // Exclude rarely-used stuff from Windows headers
#include <afxwin.h> // MFC core and standard components
#include <afxext.h> // MFC extensions
#include <afxdisp.h> // MFC OLE automation classes
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h> // MFC support for Windows Common Controls
#endif // _AFX_NO_AFXCMN_SUPPORT
#include <atlbase.h>
extern CComModule _Module;
#include <atlcom.h>
#include "OPCDA.H"
#include "OPCcomn.H"

```



```

#include "OPCError.h"

//Pre_OPC.cpp
#include "pre_opc.h"
#include "atimpl.cpp"
#define IID_DEFINED
#include "Opccomn_i.c"
#include "OPCDA_I.C"

//ASync_RWDlg.h
//

#if !defined(AFX_ASYNC_RWDLG_H__DE84F897_EBB8_483B_98F4_CFE8C7F228C5__INC
LUDED_)
#define
AFX_ASYNC_RWDLG_H__DE84F897_EBB8_483B_98F4_CFE8C7F228C5__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

////////////////////////////////////
// CASync_RWDlg dialog
class CASync_RWDlg : public CDialog
{
// Construction
public:
    CString m_WriteRes3;
    CString m_WriteRes2;
    CString m_WriteRes1;
    CString m_WriteVal3;
    double m_WriteVal2;
    int m_WriteVal1;
    CString m_ReadTs3;
    CString m_ReadTs2;
    CString m_ReadTs1;
    CString m_ReadQu3;
    CString m_ReadQu2;
    CString m_ReadQu1;
    CString m_ReadVal3;
    double m_ReadVal2;
    int m_ReadVal1;
    IOPCServer *m_pIOPCServer;
    BOOL m_ActiveCheck;
    CASync_RWDlg(CWnd* pParent = NULL); // standard constructor
// Dialog Data
//{{AFX_DATA(CASync_RWDlg)
enum { IDD = IDD_ASYNC_RW_DIALOG };
// NOTE: the ClassWizard will add data members here
//}}AFX_DATA

// ClassWizard generated virtual function overrides
//{{AFX_VIRTUAL(CASync_RWDlg)
protected:
virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
//}}AFX_VIRTUAL

```

```

// Implementation
protected:
    HICON m_hIcon;
    DWORD m_dwAdvise;
    OPCITEMDEF m_Items[3];
    IOPCItemMgt *m_pIOPCItemMgt;
    IOPCGroupStateMgt *m_pIOPCGroupStateMgt;
    IOPCAsyncIO2 *m_pIOPCAsyncIO2;
    OPCITEMRESULT *m_pItemResult;
    HRESULT *m_pErrors;
    OPCHANDLE m_GrpSrvHandle;
    // Generated message map functions
   //{{AFX_MSG(CASync_RWDlg)
    virtual BOOL OnInitDialog();
    afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    afx_msg void OnConn();
    afx_msg void OnWrite();
    afx_msg void OnRead();
    afx_msg void OnCHECKData();
    afx_msg void OnDisConn();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif
// !defined(AFX_ASYNC_RWDLG_H__DE84F897_EBB8_483B_98F4_CFE8C7F228C5__INCL
// UDED_)

//ASync_RWDlg.cpp
// ASync_RWDlg.cpp : implementation file
//
#include "stdafx.h"
#include "Pre_OPC.h"
#include "ASync_RW.h"
#include "ASync_RWDlg.h"
#include "Callback.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
// CAboutDlg dialog used for App About
#define LOCALE_ID 0x409 // Code 0x409 = ENGLISH
const LPWSTR szItemID0 = L"S7:[S7 connection_1]DB10,INT2"; // this item must be readable
and writeable in this sample
const LPWSTR szItemID1 = L"S7:[S7 connection_1]DB10,REAL4"; // this item must be
readable in this sample
const LPWSTR szItemID2 = L"S7:[S7 connection_1]DB10,STRING14.10";
class CAboutDlg : public CDialog
{
public:

```

```

        CAboutDlg();
// Dialog Data
   //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA
    // ClassWizard generated virtual function overrides
   //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
    //}}AFX_VIRTUAL
// Implementation
protected:
   //{{AFX_MSG(CAboutDlg)
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
    // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
///////////////////////////////////////////////////////////////////
// CASync_RWDlg dialog

CASync_RWDlg::CASync_RWDlg(CWnd* pParent /*=NULL*/)
: CDialog(CASync_RWDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CASync_RWDlg)
    // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
    m_WriteRes3= _T("");
    m_WriteRes2= _T("");
    m_WriteRes1= _T("");
    m_WriteVal3= _T("");
    m_WriteVal2=0.0;
    m_WriteVal1=0;
    m_ReadTs3= _T("");
    m_ReadTs2= _T("");
    m_ReadTs1= _T("");
    m_ReadQu3= _T("");
    m_ReadQu2= _T("");
    m_ReadQu1= _T("");
    m_ReadVal3=_T("");
    m_ReadVal2=0.0;

```

```

        m_ReadVal1=0;
        m_ActiveCheck=false;
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
    }
void CASync_RWDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT_ReadVal1, m_ReadVal1);
    DDX_Text(pDX, IDC_EDIT_ReadVal2, m_ReadVal2);
    DDX_Text(pDX, IDC_EDIT_ReadVal3, m_ReadVal3);
    DDX_Text(pDX, IDC_EDIT_ReadQu1, m_ReadQu1);
    DDX_Text(pDX, IDC_EDIT_ReadQu2, m_ReadQu2);
    DDX_Text(pDX, IDC_EDIT_ReadQu3, m_ReadQu3);
    DDX_Text(pDX, IDC_EDIT_ReadTS1, m_ReadTs1);
    DDX_Text(pDX, IDC_EDIT_ReadTS2, m_ReadTs2);
    DDX_Text(pDX, IDC_EDIT_ReadTS3, m_ReadTs3);
    DDX_Text(pDX, IDC_EDIT_WriteVal1, m_WriteVal1);
    DDX_Text(pDX, IDC_EDIT_WriteVal2, m_WriteVal2);
    DDX_Text(pDX, IDC_EDIT_WriteVal3, m_WriteVal3);
    DDX_Text(pDX, IDC_EDIT_WriteRes1, m_WriteRes1);
    DDX_Text(pDX, IDC_EDIT_WriteRes2, m_WriteRes2);
    DDX_Text(pDX, IDC_EDIT_WriteRes3, m_WriteRes3);
    DDX_Check(pDX, IDC_CHK_Active, m_ActiveCheck);
    //{AFX_DATA_MAP(CASync_RWDlg)
    // NOTE: the ClassWizard will add DDX and DDV calls here
    //}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CASync_RWDlg, CDialog)
    //{AFX_MSG_MAP(CASync_RWDlg)
    ON_WM_SYSCOMMAND()
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_BN_CLICKED(IDC_BUTTON_Conn, OnConn)
    ON_BN_CLICKED(IDC_BUTTON_DisConn, OnDisConn)
    ON_BN_CLICKED(IDC_BUTTON_ARead, OnRead)
    ON_BN_CLICKED(IDC_BUTTON_AWrite, OnWrite)
    ON_BN_CLICKED(IDC_CHK_Active, OnCHECKData)
    //}AFX_MSG_MAP
END_MESSAGE_MAP()
////////////////////////////////////
// CASync_RWDlg message handlers
BOOL CASync_RWDlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    // Add "About..." menu item to system menu.
    // IDM_ABOUTBOX must be in the system command range.
    ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
    ASSERT(IDM_ABOUTBOX < 0xF000);
    CMenu* pSysMenu = GetSystemMenu(FALSE);
    if (pSysMenu != NULL)
    {
        CString strAboutMenu;
        strAboutMenu.LoadString(IDS_ABOUTBOX);
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);

```

```

        pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
    }
}
// Set the icon for this dialog. The framework does this automatically
// when the application's main window is not a dialog
SetIcon(m_hIcon, TRUE);           // Set big icon
SetIcon(m_hIcon, FALSE);        // Set small icon
// TODO: Add extra initialization here
return TRUE; // return TRUE unless you set the focus to a control
}
void CASync_RWDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
    if ((nID & 0xFFF0) == IDM_ABOUTBOX)
    {
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
    }
    else
    {
        CDialog::OnSysCommand(nID, lParam);
    }
}
// If you add a minimize button to your dialog, you will need the code below
// to draw the icon. For MFC applications using the document/view model,
// this is automatically done for you by the framework.
void CASync_RWDlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // device context for painting
        SendMessage(WM_ICONERASEBKGND, (LPARAM) dc.GetSafeHdc(), 0);
        // Center icon in client rectangle
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;
        // Draw the icon
        dc.DrawIcon(x, y, m_hIcon);
    }
    else
    {
        CDialog::OnPaint();
    }
}
// The system calls this to obtain the cursor to display while the user drags
// the minimized window.
HCURSOR CASync_RWDlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}
void CASync_RWDlg::OnConn()
{
    HRESULT          r1;
    CLSID            clsid;

```

```

LONG          TimeBias = 0;
FLOAT         PercentDeadband = 0.0;
DWORD         RevisedUpdateRate;
LPWSTR        ErrorStr1,ErrorStr2,ErrorStr3;
CString       szErrorText;
m_pItemResult = NULL;
// Initialize the COM library
r1 = CoInitialize(NULL);
if (r1 != S_OK)
{
    if (r1 == S_FALSE)
    {
        MessageBox("COM Library already initialized",
                    "Error CoInitialize()", MB_OK+MB_ICONEXCLAMATION);
    }
    else
    {
        szErrorText.Format("Initialisation of COM Library failed. ErrorCode= %4x", r1);
        MessageBox(szErrorText,"Error CoInitialize()", MB_OK+MB_ICONERROR);
        SendMessage(WM_CLOSE);
        return;
    }
}
// Given a ProgID, this looks up the associated CLSID in the registry
r1 = CLSIDFromProgID(L"OPC.SimaticNET", &clsid);
if (r1 != S_OK)
{
    MessageBox("Retrival of CLSID failed",
                "Error CLSIDFromProgID()", MB_OK+MB_ICONERROR);
    CoUninitialize();
    SendMessage(WM_CLOSE);
    return;
}
// Create the OPC server object and query for the IOPCServer interface of the object
r1 = CoCreateInstance (clsid, NULL, CLSCTX_LOCAL_SERVER ,IID_IOPCServer,
(void**)&m_pIOPCServer);
if (r1 != S_OK)
{
    MessageBox("Creation of IOPCServer-Object failed",
                "Error CoCreateInstance()", MB_OK+MB_ICONERROR);
    m_pIOPCServer = NULL;
    CoUninitialize();
    SendMessage(WM_CLOSE);
    return;
}
// Add a group object "grp1" and query for interface IOPCItemMgt
r1=m_pIOPCServer->AddGroup(L"grp1",FALSE,500,1,
&TimeBias,&PercentDeadband,LOCALE_ID,&m_GrpSrvHandle,&RevisedUpdateRate,
IID_IOPCItemMgt,(LPUNKNOWN*)&m_pIOPCItemMgt);
if (r1 == OPC_S_UNSUPPORTEDRATE)
{
    szErrorText.Format ("Revised Update Rate %d is different from Requested
Update Rate 500",RevisedUpdateRate );
    AfxMessageBox (szErrorText);
}
else
    if (FAILED(r1))
    {
        MessageBox("Can't add Group to Server!",
                    "Error AddGroup()", MB_OK+MB_ICONERROR);
        m_pIOPCServer->Release();
        m_pIOPCServer = NULL;
    }
}

```

```

        CoUninitialize();
        SendMessage(WM_CLOSE);
        return;
    }
    // define item tables with 2 items as in parameters for AddItem
    m_Items[0].szAccessPath          = L"";
    m_Items[0].szItemID              = L"S7:[S7 connection_1]DB10,INT2";
    m_Items[0].bActive               = TRUE;
    m_Items[0].hClient               = 1;
    m_Items[0].dwBlobSize            = 0;
    m_Items[0].pBlob                 = NULL;
    m_Items[0].vtRequestedDataType = 2;
    m_Items[1].szAccessPath          = L"";
    m_Items[1].szItemID              = L"S7:[S7 connection_1]DB10,Real4";
    m_Items[1].bActive               = TRUE;
    m_Items[1].hClient               = 2;
    m_Items[1].dwBlobSize            = 0;
    m_Items[1].pBlob                 = NULL;
    m_Items[1].vtRequestedDataType = 5;
    m_Items[2].szAccessPath          = L"";
    m_Items[2].szItemID              = L"S7:[S7 connection_1]DB10,STRING14.10";
    m_Items[2].bActive               = TRUE;
    m_Items[2].hClient               = 3;
    m_Items[2].dwBlobSize            = 0;
    m_Items[2].pBlob                 = NULL;
    m_Items[2].vtRequestedDataType = 8;
    r1 = m_pIOPCItemMgt->AddItems(    3,m_Items,&m_pltemResult,&m_pErrors);
    if ( (r1 != S_OK) && (r1 != S_FALSE) )
    {
        MessageBox("AddItems failed!",
            "Error AddItems()", MB_OK+MB_ICONERROR);
        m_pIOPCItemMgt->Release();
        m_pIOPCItemMgt = NULL;
        m_GrpSrvHandle = 0;
        m_pIOPCServer->Release();
        m_pIOPCServer = NULL;
        CoUninitialize();
        SendMessage(WM_CLOSE);
        return;
    }
    else
    {
        m_pIOPCServer->GetErrorString(m_pErrors[0], LOCALE_ID, &ErrorStr1);
        m_pIOPCServer->GetErrorString(m_pErrors[1], LOCALE_ID, &ErrorStr2);
        m_pIOPCServer->GetErrorString(m_pErrors[2], LOCALE_ID, &ErrorStr3);
        CString szOut;
        szOut.Format("Item %ls :\n %ls\n\nItem %ls :\n %ls\n", szItemID0, ErrorStr1, szItemID1,
            ErrorStr2,szItemID2, ErrorStr3);
        MessageBox(szOut, "Result AddItems()", MB_OK+MB_ICONEXCLAMATION);
        CoTaskMemFree(ErrorStr1);
        CoTaskMemFree(ErrorStr2);
        CoTaskMemFree(ErrorStr3);
        // go on
    }
    // check if Item0 is readable and writeable
    if (m_pltemResult[0].dwAccessRights != (OPC_READABLE + OPC_WRITEABLE))
    {

```

```

        MessageBox("Item 0 ist not readable and writeable.\nThis is essential for
correct operation of this sample.", "Result AddItems()", MB_OK+MB_ICONEXCLAMATION);
    }
    // Get interface for GroupStateMgt
    r1=m_pIOPCItemMgt->QueryInterface(IID_IOPCGroupStateMgt, (void**)
&m_pIOPCGroupStateMgt);
    if (r1 != S_OK)
    {
        MessageBox("No IOPCGroupStateMgt found",
            "Error QueryInterface()", MB_OK+MB_ICONERROR);
        CoTaskMemFree(m_pltemResult);
        m_pIOPCItemMgt->Release();
        m_pIOPCItemMgt = 0;
        m_GrpSrvHandle = 0;
        m_pIOPCServer->Release();
        m_pIOPCServer = NULL;
        CoUninitialize();
        SendMessage(WM_CLOSE);
        return;
    }
    // query interface for async calls on group object
    r1 = m_pIOPCItemMgt->QueryInterface(IID_IOPCAsyncIO2,
(void**)&m_pIOPCAsyncIO2);
    if (r1 < 0)
    {
        MessageBox("No IOPCAsyncIO found!",
            "Error QueryInterface()", MB_OK+MB_ICONERROR);
        CoTaskMemFree(m_pltemResult);
        m_pIOPCItemMgt->Release();
        m_pIOPCItemMgt = 0;
        m_GrpSrvHandle = 0;
        m_pIOPCServer->Release();
        m_pIOPCServer = NULL;
        CoUninitialize();
        SendMessage(WM_CLOSE);
    }
    return;
}
// Activate Group according to Checkbox
OnCHECKData();
// Establish Callback for all async operations
CComObject<COPCDataCallback>* pCOPCDataCallback;// Pointer to Callback Object
// Create Instance of Callback Object using an ATL template
CComObject<COPCDataCallback>::CreateInstance(&pCOPCDataCallback);

// pass pointer of this dialog class to callback object
pCOPCDataCallback->InformAboutDialog(this);
// query IUnknown interface of callback object, needed for the ATL Advise
LPUNKNOWN pCbUnk;
pCbUnk = pCOPCDataCallback->GetUnknown();
// Creates a connection between the OPC servers's connection point and
// this client's sink (the callback object).
HRESULT hRes = AtlAdvise( m_pIOPCGroupStateMgt,
                        pCbUnk,IID_IOPCDataCallback,&m_dwAdvise );

if (hRes != S_OK)
{
    AfxMessageBox("Advise failed!");
    CoTaskMemFree(m_pltemResult);
    m_pIOPCItemMgt->Release();
    m_pIOPCItemMgt = 0;
}

```



```

        m_GrpSrvHandle = 0;
        m_pIOPCServer->Release();
        m_pIOPCServer = NULL;
        CoUninitialize();
        SendMessage(WM_CLOSE);
    return;
}
}
void CASync_RWDlg::OnRead()
{
    OPCHANDLE      *phServer;
    DWORD          dwCancelID;
    HRESULT        *pErrors;
    HRESULT        r1;
    CString        szOut;
    LPWSTR         ErrorStr1;
    LPWSTR         ErrorStr2;
    LPWSTR         ErrorStr3;

    if (m_pErrors[0] != S_OK || m_pErrors[1] != S_OK) // Item not available
    {
        MessageBox("OPC Item0 not available!",
                  "Error Read async", MB_OK+MB_ICONERROR);
        return;
    }
    // Memory allocation really needed, if more than 1 item to be read
    phServer = new OPCHANDLE[3];
    // Select item by server handle, received at AddItem
    phServer[0] = m_pltemResult[0].hServer;
    phServer[1] = m_pltemResult[1].hServer;
    phServer[2] = m_pltemResult[2].hServer;
    r1 = m_pIOPCAsyncIO2->Read(3, phServer, 10, &dwCancelID, &pErrors);
    delete[] phServer;
    if (r1 == S_FALSE)
    {
        m_pIOPCServer->GetErrorString(pErrors[0], LOCALE_ID, &ErrorStr1);
        m_pIOPCServer->GetErrorString(m_pErrors[1], LOCALE_ID, &ErrorStr2);
        m_pIOPCServer->GetErrorString(m_pErrors[2], LOCALE_ID, &ErrorStr3);
        CString szOut;
        szOut.Format("Item %ls : \n %ls \n \n Item %ls : \n %ls \n", szItemID0, ErrorStr1, szItemID1,
                    ErrorStr2, szItemID2, ErrorStr3);
        MessageBox(szOut, "Error Reading ", MB_OK+MB_ICONERROR);
        // release ErrorStr
        CoTaskMemFree(ErrorStr1);
        CoTaskMemFree(ErrorStr2);
        CoTaskMemFree(ErrorStr3);
    }
    if (FAILED(r1))
    {
        szOut.Format ("Method call IOPCAsyncIO2::Read failed with error code %x", r1);
        MessageBox(szOut, "Error Reading Item0", MB_OK+MB_ICONERROR);
    }
    else
    {
        // release [out] parameter in case of not failed
        CoTaskMemFree(pErrors);
    }
}
void CASync_RWDlg::OnWrite()

```

```

{
    OPCHANDLE          *phServer;
    DWORD              dwCancelID;
    VARIANT            values[2];
    HRESULT            *pErrors;
    HRESULT            r1;
    LPWSTR             ErrorStr;
    LPWSTR             ErrorStr1;
    LPWSTR             ErrorStr2;
    CString            szOut;
    if (m_pErrors[0] != S_OK || m_pErrors[1] != S_OK || m_pErrors[2] != S_OK)    {
        MessageBox("OPC Item not available!",
            "Error Write async", MB_OK+MB_ICONERROR);
        return;
    }
    // Select item by server handle, received at AddItem
    phServer = new OPCHANDLE[3];
    phServer[0] = m_pltemResult[0].hServer;
    phServer[1] = m_pltemResult[1].hServer;
    phServer[2] = m_pltemResult[2].hServer;
    // Retrieve data from dialog
    UpdateData(TRUE);
    // Set Variant with datatype and received value
    values[0].vt = VT_I2;
    values[0].iVal = m_WriteVal1;
    values[1].vt = VT_R8;
    values[1].dblVal = m_WriteVal2;
    values[2].vt = VT_BSTR;
    //values[2].bstrVal = m_WriteVal3;
    r1 = m_pIOPCAsyncIO2->Write(2, values, 2,
        &dwCancelID
        &pErrors
        );
    delete[] phServer;

    if (r1 == S_FALSE)
    {
        m_pIOPCServer->GetErrorString(pErrors[0], LOCALE_ID, &ErrorStr);
        m_pIOPCServer->GetErrorString(pErrors[1], LOCALE_ID, &ErrorStr1);
        m_pIOPCServer->GetErrorString(pErrors[2], LOCALE_ID, &ErrorStr2);
        m_WriteRes1 = ErrorStr;
        m_WriteRes1.Remove('\r');
        m_WriteRes1.Remove('\n');
        m_WriteRes2 = ErrorStr1;
        m_WriteRes2.Remove('\r');
        m_WriteRes2.Remove('\n');
        m_WriteRes3 = ErrorStr2;
        m_WriteRes3.Remove('\r');
        m_WriteRes3.Remove('\n');
        UpdateData(FALSE);
        // release parameter in case of S_FALSE
        CoTaskMemFree(ErrorStr);
        CoTaskMemFree(ErrorStr1);
        CoTaskMemFree(ErrorStr2);
    }

    if (FAILED(r1))
    {szOut.Format ("Method call IOPCAsyncIO2::Write failed with error code %x", r1);
        MessageBox(szOut, "Error Writing Item0", MB_OK+MB_ICONERROR);
    }
}

```

```

    }
    else
    {
        // release [out] parameter in case of not failed
        CoTaskMemFree(pErrors);
    }
}
void CASync_RWDlg::OnDisConn()
{
    HRESULT          r1;
    OPCHANDLE        *phServer;
    HRESULT          *pErrors;
    LPWSTR           ErrorStr;
    char             str[100];
    phServer = new OPCHANDLE[3];
    phServer[0] = m_pltemResult[0].hServer;
    phServer[1] = m_pltemResult[1].hServer;
    phServer[2] = m_pltemResult[2].hServer;
    r1 = m_pIOPCItemMgt->RemoveItems(3, phServer,      &pErrors);

    if ( (r1 != S_OK) && (r1 != S_FALSE) )
        {
            MessageBox("RemoveItems failed!",
                "Error RemoveItems()", MB_OK+MB_ICONERROR);
        }
    else
        {
            m_pIOPCServer->GetErrorString(pErrors[0], LOCALE_ID, &ErrorStr);
            sprintf(str, "%ls\n", ErrorStr);
            MessageBox(str, "Result RemoveItems()", MB_OK+MB_ICONEXCLAMATION);
            CoTaskMemFree(ErrorStr);
        }

    delete[] phServer;
    CoTaskMemFree(pErrors);
    CoTaskMemFree(m_pltemResult);
    m_pltemResult=NULL;
    CoTaskMemFree(m_pErrors);
    m_pErrors = NULL;
    // Release interface for sync calls
    m_pIOPCAsyncIO2->Release();
    m_pIOPCAsyncIO2 = NULL;
    // Release ItemManagement interface
    m_pIOPCItemMgt->Release();
    m_pIOPCItemMgt = NULL;
    // Remove Group
    r1=m_pIOPCServer->RemoveGroup(m_GrpSrvHandle, TRUE);
    if (r1 != S_OK)
        {
            MessageBox("RemoveGroup failed!",
                "Error RemoveGroup()", MB_OK+MB_ICONERROR);
        }
    m_GrpSrvHandle = NULL;
    // Release OPC-Server
    m_pIOPCServer->Release();
    m_pIOPCServer = NULL;
    // Uninit COM
    CoUninitialize();
}
void CASync_RWDlg::OnCHECKData()
{

```

```

// TODO: Add your control notification handler code here
UpdateData(TRUE);
DWORD dwRevUpdateRate;
int aa=1;
HRESULT r1= m_pIOPCGroupStateMgt->SetState(NULL,
&dwRevUpdateRate, &m_ActiveCheck, NULL,, NULL,NULL, NULL);
if (FAILED(r1))
{
    MessageBox("Set State failed",
    "Error", MB_OK+MB_ICONERROR);
    return;
}
}

```

//callback.h

```

class COPCDataCallback :
public CComObjectRoot,
    public IOPCDataCallback
{
public:
COPCDataCallback() {};
BEGIN_COM_MAP(COPCDataCallback)
    COM_INTERFACE_ENTRY(IOPCDataCallback)
END_COM_MAP()
// IOPCDataCallback
    STDMETHODCALLTYPE OnDataChange(
        /* [in] */ DWORD dwTransid,
        /* [in] */ OPCHANDLE hGroup,
        /* [in] */ HRESULT hrMasterquality,
        /* [in] */ HRESULT hrMastererror,
        /* [in] */ DWORD dwCount,
        /* [size_is][in] */ OPCHANDLE __RPC_FAR *phClientItems,
        /* [size_is][in] */ VARIANT __RPC_FAR *pvValues,
        /* [size_is][in] */ WORD __RPC_FAR *pwQualities,
        /* [size_is][in] */ FILETIME __RPC_FAR *pftTimeStamps,
        /* [size_is][in] */ HRESULT __RPC_FAR *pErrors);

    STDMETHODCALLTYPE OnReadComplete(
        /* [in] */ DWORD dwTransid,
        /* [in] */ OPCHANDLE hGroup,
        /* [in] */ HRESULT hrMasterquality,
        /* [in] */ HRESULT hrMastererror,
        /* [in] */ DWORD dwCount,
        /* [size_is][in] */ OPCHANDLE __RPC_FAR *phClientItems,
        /* [size_is][in] */ VARIANT __RPC_FAR *pvValues,
        /* [size_is][in] */ WORD __RPC_FAR *pwQualities,
        /* [size_is][in] */ FILETIME __RPC_FAR *pftTimeStamps,
        /* [size_is][in] */ HRESULT __RPC_FAR *pErrors);
    STDMETHODCALLTYPE OnWriteComplete(
        /* [in] */ DWORD dwTransid,
        /* [in] */ OPCHANDLE hGroup,
        /* [in] */ HRESULT hrMastererr,
        /* [in] */ DWORD dwCount,
        /* [size_is][in] */ OPCHANDLE __RPC_FAR *pClienthandles,
        /* [size_is][in] */ HRESULT __RPC_FAR *pErrors);

```

```

STDMETHODIMP OnCancelComplete(
    /* [in] */ DWORD dwTransid,
    /* [in] */ OPCHANDLE hGroup)
    {
        return S_OK;
    };

void InformAboutDialog (CASync_RWDlg *pCDlgClass)
    {m_pCDlgClass = pCDlgClass;}

protected:
    CASync_RWDlg *m_pCDlgClass;
};
//callback.cpp
#include "stdafx.h"
#include "pre_opc.h"          #include "ASync_RW.h" // include class declaration for
application class
#include "ASync_RWDlg.h" // include class declaration of dialog class
#include "Callback.h"        // for OPC: include class declaration fpr this callback class
CString GetQualityText(UINT qnr);
#define LOCALE_ID 0x409 // Code 0x409 = ENGLISH
                        // text description from quality code
////////////////////////////////////
//
// IOPCDataCallback - implementation

// onDataChange called by OPC server when the OPC server has detected modified data
// (being called only when group and item is set to active)
STDMETHODIMP COPCDataCallback::OnDataChange(
    /* [in] */ DWORD dwTransid,
    /* [in] */ OPCHANDLE hGroup,
    /* [in] */ HRESULT hrMasterquality,
    /* [in] */ HRESULT hrMastererror,
    /* [in] */ DWORD dwCount,
    /* [size_is][in] */ OPCHANDLE __RPC_FAR *phClientItems,
    /* [size_is][in] */ VARIANT __RPC_FAR *pvValues,
    /* [size_is][in] */ WORD __RPC_FAR *pwQualities,
    /* [size_is][in] */ FILETIME __RPC_FAR *pftTimeStamps,
    /* [size_is][in] */ HRESULT __RPC_FAR *pErrors)
{
    DWORD i;
    for (i = 0; i<dwCount; i++)
    {
        switch (phClientItems[i])
        {
            case 1:
                m_pCDlgClass->m_ReadVal1 =pvValues[i].iVal;
                m_pCDlgClass->m_ReadQu1 =GetQualityText(pwQualities[0]);
                m_pCDlgClass->m_ReadTs1
                =COleDateTime( pftTimeStamps[0] ).Format();
                break;
            case 2:
                m_pCDlgClass->m_ReadVal2 =pvValues[i].dblVal;
                m_pCDlgClass->m_ReadQu2 =GetQualityText(pwQualities[1]);
                m_pCDlgClass->m_ReadTs2
                =COleDateTime( pftTimeStamps[1] ).Format();

```

```

        break;
        case 3:
            m_pCDlgClass->m_ReadVal3 =pvValues[i].bstrVal;
            m_pCDlgClass->m_ReadQu3 =GetQualityText(pwQualities[2]);
            m_pCDlgClass->m_ReadTs3
=COleDateTime( pftTimeStamps[2] ).Format();
            break;
        }
        m_pCDlgClass->UpdateData(FALSE);
    }
    return S_OK;
};
STDMETHODIMP COPCDataCallback::OnReadComplete(
    /* [in] */ DWORD dwTransid,
    /* [in] */ OPCHANDLE hGroup,
    /* [in] */ HRESULT hrMasterquality,
    /* [in] */ HRESULT hrMastererror,
    /* [in] */ DWORD dwCount,
    /* [size_is][in] */ OPCHANDLE __RPC_FAR *phClientItems,
    /* [size_is][in] */ VARIANT __RPC_FAR *pvValues,
    /* [size_is][in] */ WORD __RPC_FAR *pwQualities,
    /* [size_is][in] */ FILETIME __RPC_FAR *pftTimeStamps,
    /* [size_is][in] */ HRESULT __RPC_FAR *pErrors)
{
    if (pErrors[0] == S_OK)
    {
        m_pCDlgClass->m_ReadVal1 =pvValues[0].iVal;
        m_pCDlgClass->m_ReadQu1 =GetQualityText(pwQualities[0]);
        m_pCDlgClass->m_ReadTs1
=COleDateTime( pftTimeStamps[0] ).Format();

        //delete pvarOut;
    }
    else {
        /* error case: display error text */
        CString readQuality = GetQualityText(pErrors[0]);
        m_pCDlgClass->m_ReadQu1 =readQuality;
    }
    if (pErrors[1] == S_OK)
    {
        m_pCDlgClass->m_ReadVal2 =pvValues[1].dblVal;
        m_pCDlgClass->m_ReadQu2 =GetQualityText(pwQualities[1]);
        m_pCDlgClass->m_ReadTs2
=COleDateTime(pftTimeStamps[1]).Format();

        //delete pvarOut;
    }
    else
    {
        /* error case: display error text */
        CString readQuality1 = GetQualityText(pErrors[1]);
        m_pCDlgClass->m_ReadQu2 =readQuality1;
    }
    if (pErrors[2] == S_OK)
    {

```

```

        m_pCDlgClass->m_ReadVal3 =pvValues[2].bstrVal;
        m_pCDlgClass->m_ReadQu3 =GetQualityText(pwQualities[2]);
        m_pCDlgClass-
>m_ReadTs2=COleDateTime( pftTimeStamps[2] ).Format();

        //delete pvarOut;
    }
    else
    {
        /* error case: display error text */
        CString readQuality2 = GetQualityText(pErrors[2]);
        m_pCDlgClass->m_ReadQu3 =readQuality2;
    }
    m_pCDlgClass->UpdateData(FALSE);
    return S_OK;
};

STDMETHODIMP COPCDataCallback::OnWriteComplete(
    /* [in] */ DWORD dwTransid,
    /* [in] */ OPCHANDLE hGroup,
    /* [in] */ HRESULT hrMastererr,
    /* [in] */ DWORD dwCount,
    /* [size_is][in] */ OPCHANDLE __RPC_FAR *pClienthandles,
    /* [size_is][in] */ HRESULT __RPC_FAR *pErrors)
{
    LPWSTR          ErrorStr1;
    m_pCDlgClass->m_pIOPCServer->GetErrorString(pErrors[0], LOCALE_ID, &ErrorStr1);
    m_pCDlgClass->m_WriteRes1=ErrorStr1;
    m_pCDlgClass->m_WriteRes1.Remove("\r");
    m_pCDlgClass->m_WriteRes1.Remove("\n");
    CoTaskMemFree(ErrorStr1);
    m_pCDlgClass->UpdateData(FALSE);
    return S_OK;
};

CString GetQualityText(UINT qnr)
{
    CString qstr;
    switch(qnr)
    {
        case OPC_QUALITY_BAD:                qstr = "BAD";
            break;
        case OPC_QUALITY_UNCERTAIN:         qstr = "UNCERTAIN";
            break;
        case OPC_QUALITY_GOOD:              qstr = "GOOD";
            break;
        case OPC_QUALITY_NOT_CONNECTED:     qstr = "NOT_CONNECTED";
            break;
        case OPC_QUALITY_DEVICE_FAILURE:    qstr = "DEVICE_FAILURE";
            break;
        case OPC_QUALITY_SENSOR_FAILURE:    qstr = "SENSOR_FAILURE";
            break;
        case OPC_QUALITY_LAST_KNOWN:        qstr = "LAST_KNOWN";
            break;
        case OPC_QUALITY_COMM_FAILURE:      qstr = "COMM_FAILURE";
            break;
        case OPC_QUALITY_OUT_OF_SERVICE:    qstr = "OUT_OF_SERVICE";
            break;
        case OPC_QUALITY_LAST_USABLE:       qstr = "LAST_USABLE";
    }
}

```

```
        break;
    case OPC_QUALITY_SENSOR_CAL:    qstr = "SENSOR_CAL";
        break;
    case OPC_QUALITY_EGU_EXCEEDED: qstr = "EGU_EXCEEDED";
        break;
    case OPC_QUALITY_SUB_NORMAL:    qstr = "SUB_NORMAL";
        break;
    case OPC_QUALITY_LOCAL_OVERRIDE:qstr = "LOCAL_OVERRIDE";
        break;
    default:                        qstr = "UNKNOWN ERROR";
    }
    return qstr;
}
```

附录一 推荐网址**自动化系统**

西门子（中国）有限公司

工业自动化与驱动技术集团 客户服务与支持中心

网站首页: www.4008104288.com.cn

自动化系统 下载中心:

<http://www.ad.siemens.com.cn/download/DocList.aspx?TypeId=0&CatFirst=1>

自动化系统 全球技术资源:

<http://support.automation.siemens.com/CN/view/zh/10805045/130000>

“找答案” 自动化系统版区:

<http://www.ad.siemens.com.cn/service/answer/category.asp?cid=1027>

通信/网络

西门子（中国）有限公司

工业自动化与驱动技术集团 客户服务与支持中心

网站首页: www.4008104288.com.cn

通信/网络 下载中心:

<http://www.ad.siemens.com.cn/download/DocList.aspx?TypeId=0&CatFirst=12>

通信/网络 全球技术资源:

<http://support.automation.siemens.com/CN/view/zh/10805868/130000>

“找答案” Net版区:

<http://www.ad.siemens.com.cn/service/answer/category.asp?cid=1031>

注意事项

应用示例与所示电路、设备及任何可能结果没有必然联系，并不完全相关。应用示例不表示客户的具体解决方案。它们仅对典型应用提供支持。用户负责确保所述产品的正确使用。这些应用示例不能免除用户在确保安全、专业使用、安装、操作和维护设备方面的责任。当使用这些应用示例时，应意识到西门子不对在所述责任条款范围之外的任何损坏/索赔承担责任。我们保留随时修改这些应用示例的权利，恕不另行通知。如果这些应用示例与其它西门子出版物(例如，目录)给出的建议不同，则以其它文档的内容为准。

声明

我们已核对过本手册的内容与所描述的硬件和软件相符。由于差错难以完全避免，我们不能保证完全一致。我们会经常对手册中的数据进行检查，并在后续的版本中进行必要的更正。欢迎您提出宝贵意见。

版权© 西门子（中国）有限公司 2001-2008 版权保留

复制、传播或者使用该文件或文件内容必须经过权利人书面明确同意。侵权者将承担权利人的全部损失。权利人保留一切权利，包括复制、发行，以及改编、汇编的权利。

西门子（中国）有限公司