SIEMENS

# 如何在 C#中实现 OPC 数据访问

OPC data access by C#

| Getting-started | Edition (2009 年 05 月) |
| --- | --- |

**摘　要**　本文主要讲述了在 C#语言环境下，编程实现利用 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、C#、OPC、WinCC

**Key Words** SimaticNet、C#、OPC、WinCC

# 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。

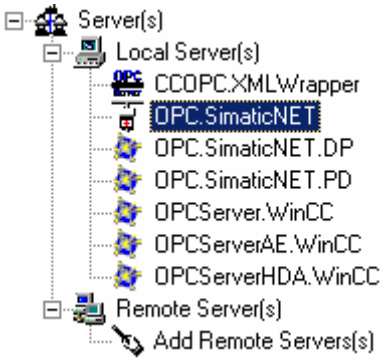



图 1：SimatcicNet 提供的 OPCServer

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：

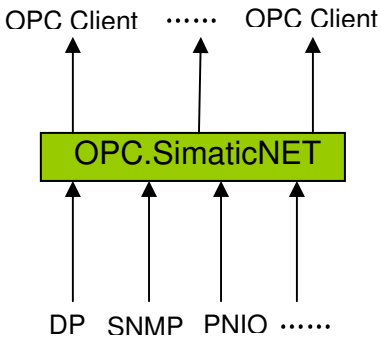



图 2：多种数据提供方式

### 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。

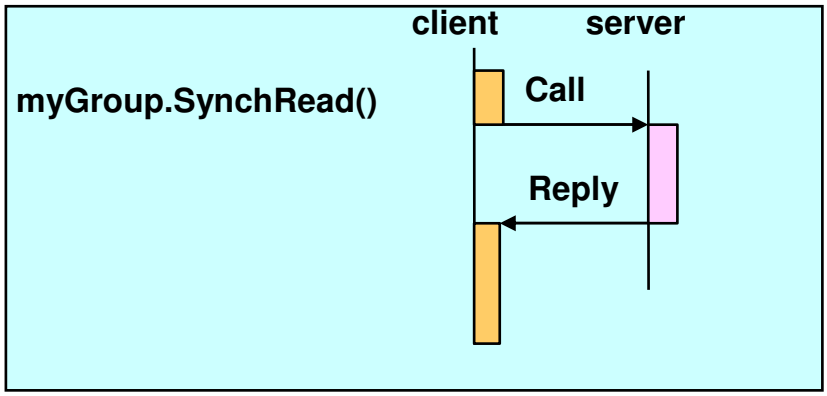



图 3　OPC 同步读写服务器-客户端数据流图

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。

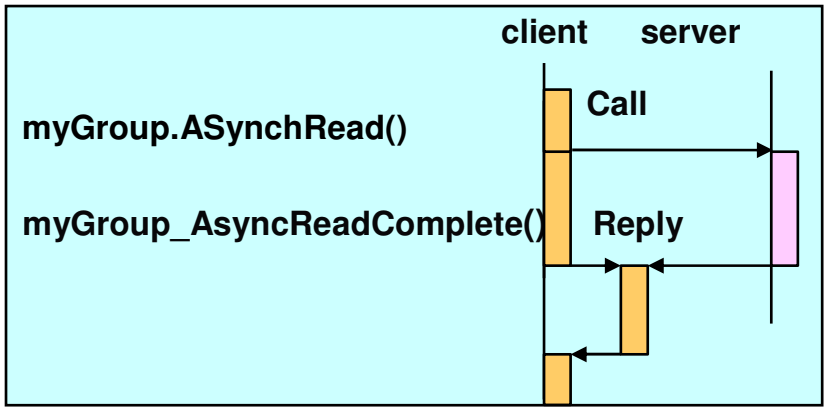



图 4　OPC 异步读服务器-客户端数据流图

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。

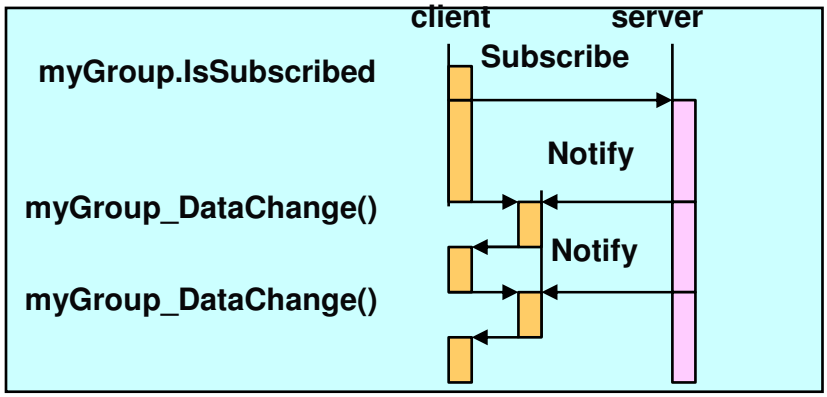



图 5　OPC 同步读服务器-客户端数据流图

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。

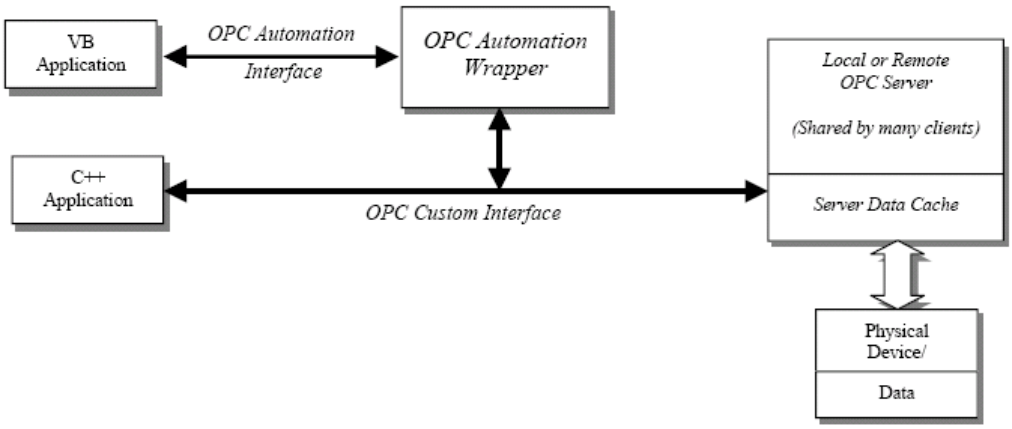



图 6　自定义接口和自动化接口

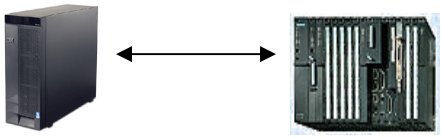
许多 OPC 服务器，包括 OPC.SimaticNet，是在 COM 平台开发的，从而对于基于.NET 框架下的 C#语言，作为客户端程序语言访问 OPCServer，需要解决两个平台间无缝迁移的问题。OPC 基金会对会员提供了 OpcRcw 动态链接库，OPC NET COM 包装器和 OPC NET API，将 OPC 复杂的规范封状成简单易用的 C#类 ，可以比较容易地实现数据访问。

本文中通过实验，逐步讲解了通过 C#编写客户端程序，访问 OPC.SimaticNet，对 PLC 数据进行读写的实现过程。自定义接口及自动化接口都进行了测试，但基于 C#的语言特性，建议采用自定义接口访问，同时有很多 OPCServer 服务商，对外是不提供自动化接口的，西门子的 SimaticNet 及 WinCC 的 OPCServer 都提供自动化接口。

## 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

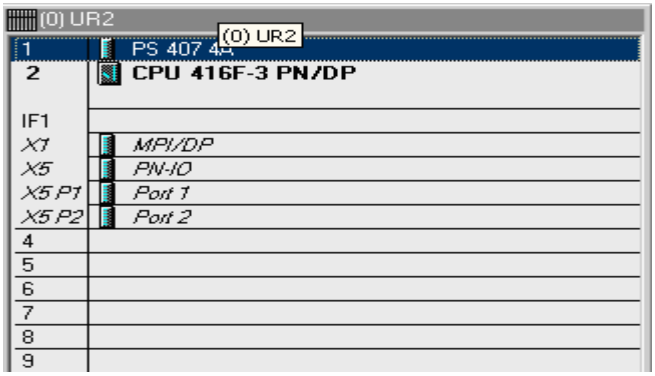
CPU：CPU414-3PN ------416-3FR05-0AB0--------192.168.0.1

### 2.2 软件要求

computer：

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

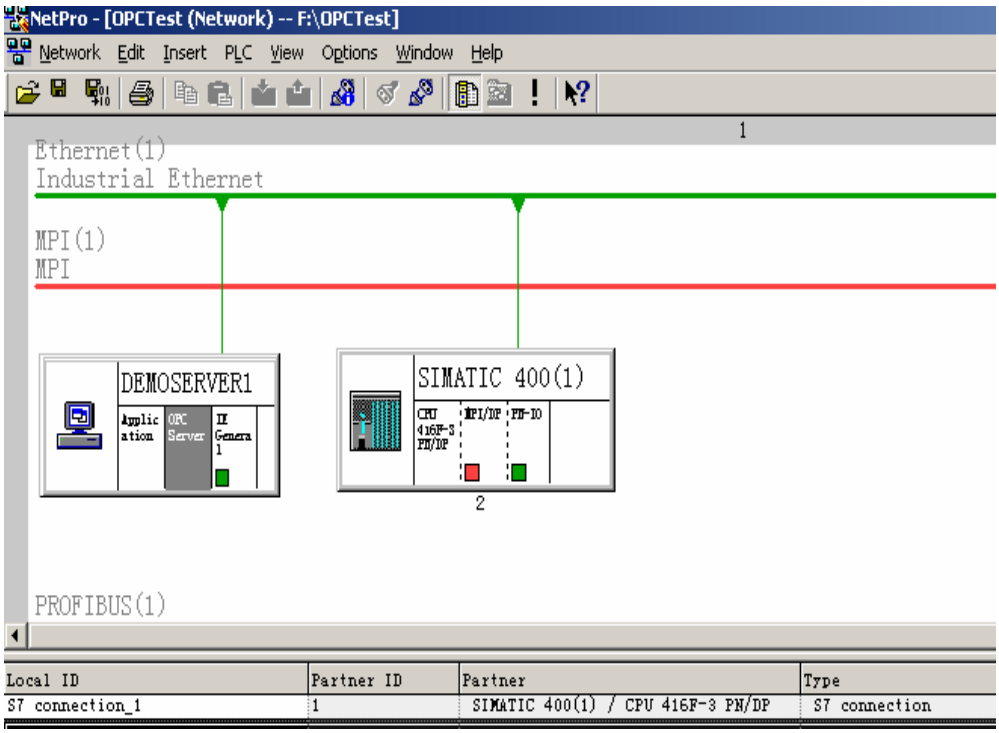
## 3、OPC Server 端组态配置



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。

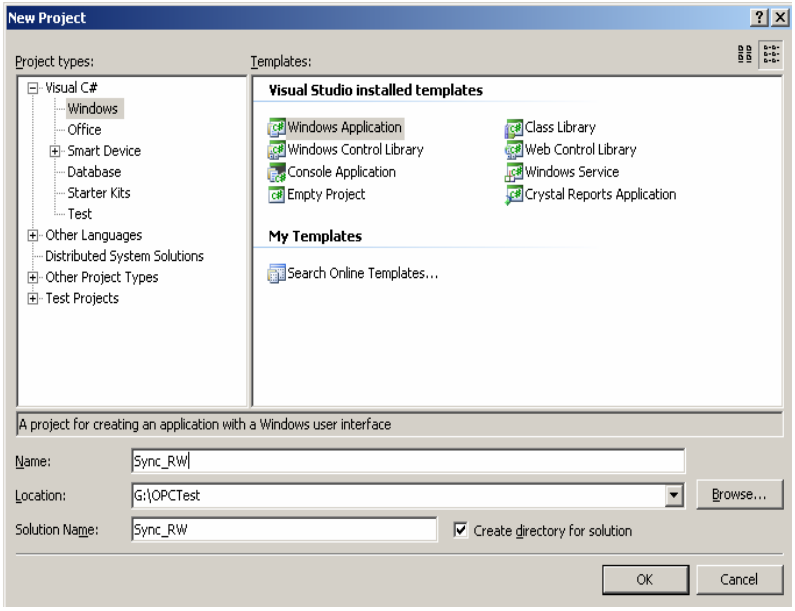
| | Item Names | Value | Format | Type | Access | Quality | e Stamp (L |
|---|---|---|---|---|---|---|---|
| 1 | S7:[S7 connection_1]DB10,INT0 | 2 | Original | int16 | RW | good | 03/18/2009 |
| 2 | S7:[S7 connection_1]DB10,INT2 | 4 | Original | int16 | RW | good | 03/18/2009 |
| 3 | S7:[S7 connection_1]DB10,REAL4 | 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、采用自定义接口过程

### 4.1 同步读写

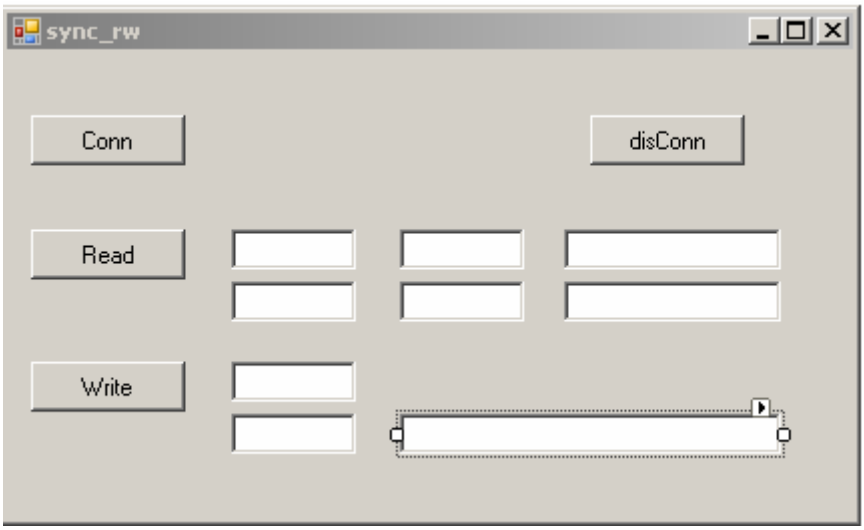
建立同步读写项目：Sync_RW




测试中，对 db10.dbw0 及 db10.dbw2 读写操作，在 Form 窗口做如下设计：

| Control | name | Text |
|---|---|---|
| Button: | Btn_Conn | Conn |
| Button: | Btn_Read | Read |
| Button: | Btn_Write | Write |
| Button: | Btn_DisConn | disConn |

TextBox: Txt_R1_Value
TextBox: Txt_R1_Quality
TextBox: Txt_R1_TimeStamp
TextBox: Txt_R2_Value
TextBox: Txt_R2_Quality
TextBox: Txt_R2_TimeStamp
TextBox: Txt_W1
TextBox: Txt_W2
TextBox: Txt_WriteStatus




第一步，添加下面命名空间：（首先需要在项目中添加相应的引用）

```
using OpcRcw.Comn;
using OpcRcw.Da;
```

第二步，定义 OPC 相关变量，

```
OpcRcw.Da.IOPCServer SrverObj;//定义OPCServer 对象
OpcRcw.Da.IOPCSyncIO IOPCSyncIO2Obj = null;//同步读对象
OpcRcw.Da.IOPCGroupStateMgt IOPCGroupStateMgtObj = null;//管理OPCGroup组对象

internal const int LOCALE_ID = 0x407;//OPCServer语言码-英语

Object MyobjGroup1 = null;//OPCGroup对象
int[] ItemServerHandle;//Item句柄数组
int pSvrGroupHandle = 0;//OPCGroup 句柄
```

第三步，连接 OPCServer，建立相应 OPCGroup 组，并添加需要读写的 Item

```
private void Btn_Conn_Click(object sender, System.EventArgs e)
{
    ......
    //定义变量
```

```
svrComponenttyp = Type.GetTypeFromProgID("OPC.SimaticNet", "192.168.0.102");
ServerObj = (OpcRcw.Da.IOPCServer)Activator.CreateInstance(svrComponenttyp);
        //"OPC.SimaticNet", "192.168.0.102"是 OPCServer 名称及所在 computer 地址
        // CreateInstance 创建一个 OPCSerer 的实例
ServerObj.AddGroup(……)//增加相应的组，定义组的特性，并输出组的句柄
IOPCSyncIO2Obj = (IOPCSyncIO)MyobjGroup1;
                        //为组同步读写定义句柄
IOPCGroupStateMgtObj = (IOPCGroupStateMgt)MyobjGroup1; //组管理对象
ItemArray[0].szAccessPath = "";
ItemArray[0].szItemID = "S7:[S7 connection_1]DB10,INT0";
                        //地址，不同数据类型表示方法不同
ItemArray[0].bActive = 1;//是否激活
ItemArray[0].hClient = 1;//标示ID，不同的Item不一样
ItemArray[0].dwBlobSize = 0;
ItemArray[0].pBlob = IntPtr.Zero;
ItemArray[0].vtRequestedDataType = 2;
……
 ((OpcRcw.Da.IOPCItemMgt)MyobjGroup1).AddItems(2, ItemArray, out pResults,
                                out pErrors); //将定义的 OPCTtem 加入组内，注意数量
……
}
```

这里需要注意两个地方，对于 hClient 每个 Item 是不一样的。

根据读写的数据类型，需更改 vtRequestedDataType 的值，具体区分在后面说明。

第四步，同步读数据

```
private void Btn_Read_Click(object sender, EventArgs e)
{
    IOPCSyncIO2Obj.Read(OPCDATASOURCE.OPC_DS_DEVICE, 2, ItemServerHandle,
                out pItemValues, out pErrors);//读数据
                ……
    Txt_R1_Value.Text = String.Format("{0}", pItemState[0].vDataValue);//读值
    Txt_R1_Quality.Text = GetQuality(pItemState[0].wQuality);//质量码
    DateTime dt = ToDateTime(pItemState[0].ftTimeStamp);
    Txt_R1_TimeStamp.Text = dt.ToString();//读取时间
}
```

在这里要注意 pItemValues 返回指向值信息的指针，要通过 OPCITEMSTATE[] pItemState 获得信息，其中 OPCITEMSTATE 是一个结构体，包含值，质量码，时间等。

```
public struct OPCITEMSTATE
 {
        public FILETIME ftTimeStamp;
        public int hClient;
```

```
        public object vDataValue;
        public short wQuality;
        public short wReserved;
    }
```

第五步，同步写数据

```
private void Btn_Write_Click(object sender, EventArgs e)
{
        ......
        IOPCSyncIO2Obj.Write(2, ItemServerHandle, values, out pErrors);
        ......
}
```

这里注意，如果数据类型不正确，数据是不能正确写入的。

第六步，注销相应实例

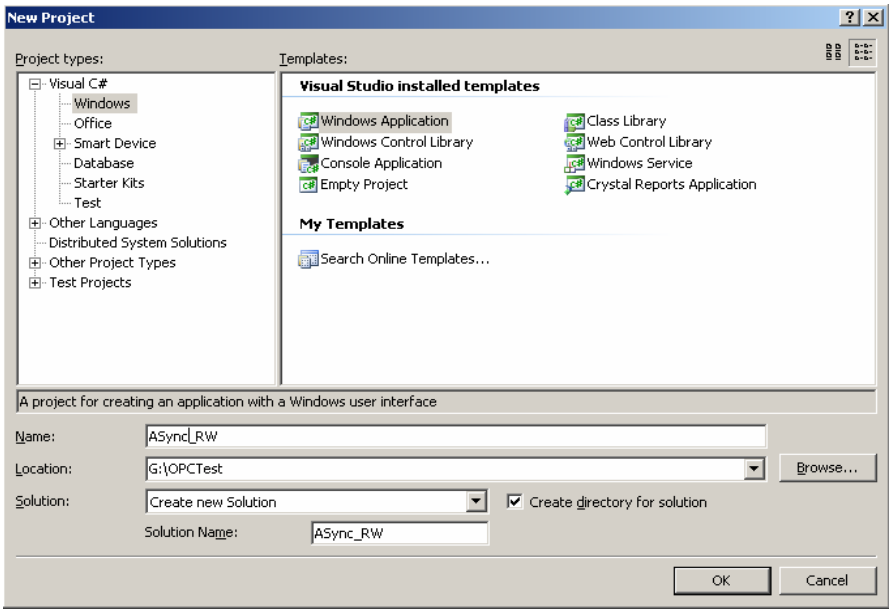
```
private void Btn_Disconn_Click(object sender, EventArgs e)
{
        ......
}
```

**参考附录中 Sync_RW 例程。**

## 4.2 异步读写

注意，订阅也是 异步方式。

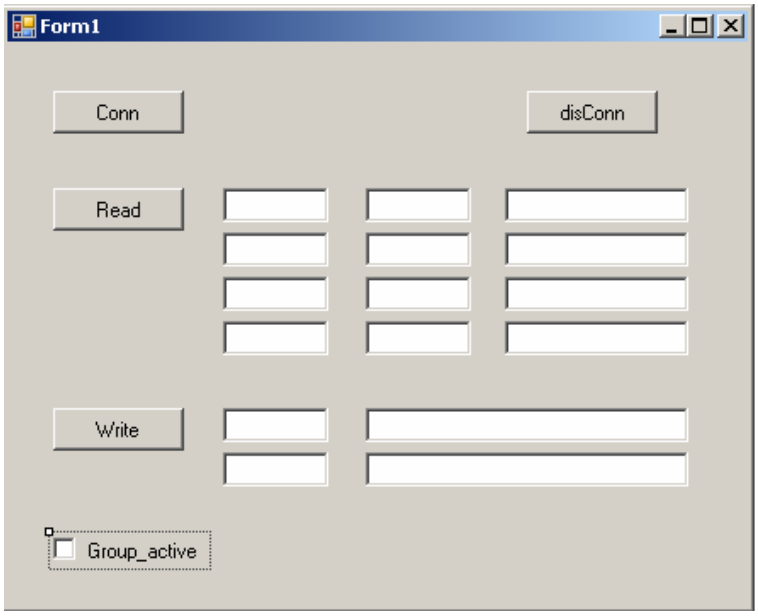
建立异步读写项目




测试中，对 db10.dbw0 及 db10.dbw2 读写操作，在 Form 窗口做如下设计：

| Control | name | Text |
|---|---|---|

Button: Btn_Conn Conn
Button: Btn_Read Read
Button: Btn_Write Write
Button: Btn_DisConn disConn
TextBox: Txt_R1_Value
TextBox: Txt_R1_Quality
TextBox: Txt_R1_TimeStamp
TextBox: Txt_R2_Value
TextBox: Txt_R2_Quality
TextBox: Txt_R2_TimeStamp
TextBox: Txt_R3_Value
TextBox: Txt_R3_Quality
TextBox: Txt_R3_TimeStamp
TextBox: Txt_R4_Value
TextBox: Txt_R4_Quality
TextBox: Txt_R4_TimeStamp
TextBox: Txt_W1
TextBox: Txt_W2
TextBox: Txt_WriteStatus
CheckBox: CHK_Btn



第一步，添加下面命名空间：（首先需要在项目中添加相应的引用）

```
using OpcRcw.Comn;
using OpcRcw.Da;
```

第二步，定义 OPC 相关变量

```
OpcRcw.Da.IOPCServer SrverObj;//定义OPCServer 对象
OpcRcw.Da.IOPCAsyncIO2 IOPCAsyncIO2Obj = null;//异步读对象
OpcRcw.Da.IOPCGroupStateMgt IOPCGroupStateMgtObj = null;//管理OPCGroup组对象

IConnectionPointContainer pIConnectionPointContainer = null;//异步事件点
IConnectionPoint pIConnectionPoint = null;//
internal const int LOCALE_ID = 0x407;//OPCServer语言码-英语

Object MyobjGroup1 = null;//OPCGroup对象
int[] ItemServerHandle;//Item句柄数组
int pSvrGroupHandle = 0;//OPCGroup 句柄

Int32 dwCookie = 0; //this client's sink
```

第三步，连接 OPCServer，建立相应 OPCGroup 组，并添加需要读写的 Item

```
private void Btn_Conn_Click(object sender, System.EventArgs e)
{
    ……

    //定义变量

 svrComponenttyp = Type.GetTypeFromProgID("OPC.SimaticNet", "192.168.0.102");
 ServerObj = (OpcRcw.Da.IOPCServer)Activator.CreateInstance(svrComponenttyp);

        //"OPC.SimaticNet", "192.168.0.102"是 OPCServer 名称及所在 computer 地址

        // CreateInstance 创建一个 OPCSerer 的实例

 ServerObj.AddGroup(……)//增加相应的组，定义组的特性，并输出组的句柄

 IOPCAsyncIO2Obj = (IOPCAsyncIO2)MyobjGroup1;
                        //为组异步读写定义句柄
 IOPCGroupStateMgtObj = (IOPCGroupStateMgt)MyobjGroup1; //组管理对象
```

与同步不同，考虑增加如下语句：

```
  pIConnectionPointContainer = (IConnectionPointContainer)MyobjGroup1;
                        //定义特定组的异步调用连接
  Guid iid = typeof(IOPCDataCallback).GUID;
                        // 为所有的异步调用创建回调
pIConnectionPointContainer.FindConnectionPoint(ref iid, out pIConnectionPoint);
                       // 为OPC Server的连接点与客户端接收点之间建立连接
 pIConnectionPoint.Advise(this, out dwCookie);

 ItemArray[0].szAccessPath = "";
 ItemArray[0].szItemID = "S7:[S7 connection_1]DB10,INT0";
                        //地址，不同数据类型表示方法不同
 ItemArray[0].bActive = 1;//是否激活
 ItemArray[0].hClient = 1;//标示ID，不同的Item不一样
 ItemArray[0].dwBlobSize = 0;
 ItemArray[0].pBlob = IntPtr.Zero;
```

```
ItemArray[0].vtRequestedDataType = 2;
……
((OpcRcw.Da.IOPCItemMgt)MyobjGroup1).AddItems(4, ItemArray, out pResults,
                                    out pErrors); //将定义的 OPCTtem 加入组内，注意数量
……
}
```

这里同样需要注意两个地方，对于 hClient 每个 Item 是不一样的。

根据读写的数据类型，需更改 vtRequestedDataType 的值，定义如上文。

另外，要注意理解异步调用时的服务器与客户端反馈关系。

第四步，异步读数据方式

```
private void btn_Read_A_Click(object sender, System.EventArgs e)
{
        ……
        IOPCAsyncIO2Obj.Read(4,ItemServerHandle,2,out nCancelid,out pErrors);
                //异步读，nCancelid、dwTransactionID 都是为了客户端服务器的对应
        ……
}
```

调用异步读回调函数

```
public virtual void OnReadComplete( System.Int32 dwTransid ,
        System.Int32 hGroup ,
        System.Int32 hrMasterquality ,
        System.Int32 hrMastererror ,
        System.Int32 dwCount ,
        int[] phClientItems , //读数据句柄
        object[] pvValues , //返回值
        short[] pwQualities , //返回质量码
        OpcRcw.Da.FILETIME[] pftTimeStamps , //返回时间戳
        int[] pErrors ) //错误码
{
        ……
        Txt_R1_Value.Text = String.Format("{0}", pvValues[0]);
        Txt_R1_Quality.Text = GetQuality(pwQualities[0]);
        DateTime dt = ToDateTime(pftTimeStamps[0]);
        Txt_R1_TimeStamp.Text = dt.ToString();
        ……
}
```

编译执行，程序会报错。

第五步，订阅方式读回调函数及实现 COM 映射

上面程序会有以下几种情况的报错：

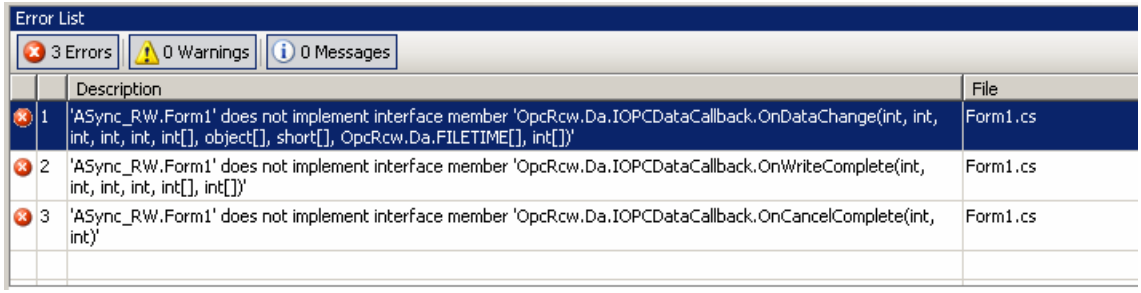
问题 1：程序执行后，弹出如下错误，Add group 报错


主要原因是 Form 要使用 IOPCDataCallback，目的是将 OPC 的接口与实现类结合在一起，实现 COM 的映射。

需要做的处理是：

```
public partial class Form1 : Form, IOPCDataCallback
```

问题 2：添加 IOPCDataCallback 接口后




主要原因是，IOPCDataCallback 有 4 个纯虚函数，必须实现

```
public virtual void OnReadComplete(System.Int32 dwTransid, //异步读完成触发
  System.Int32 hGroup,
  System.Int32 hrMasterquality,
  System.Int32 hrMastererror,
  System.Int32 dwCount,
  int[] phClientItems,
  object[] pvValues,
  short[] pwQualities,
  OpcRcw.Da.FILETIME[] pftTimeStamps,
  int[] pErrors)


public virtual void OnWriteComplete ( System.Int32 dwTransid , //异步写完成触发
  System.Int32 hGroup ,
  System.Int32 hrMastererr ,
  System.Int32 dwCount ,
  int[] pClienthandles ,
          int[] pErrors )


public virtual void OnCancelComplete(System.Int32 dwTransid, System.Int32 hGroup)
                                                          //取消特定操作触发
 public virtual void OnDataChange(Int32 dwTransid,  //订阅方式下读触发
    Int32 hGroup,
```

```
            Int32 hrMasterquality,
             Int32 hrMastererror,
            Int32 dwCount,
            int[] phClientItems,
            object[] pvValues,
            short[] pwQualities,
            OpcRcw.Da.FILETIME[] pftTimeStamps,
            int[] pErrors)
```

问题 3：运行时，有时会弹出 Cross-thread operation not valid 错误，这是 C#中对控件继承性的一种严格要求，在调试时会出现，可以做如下处理。

在 Form 的.ctor 中，InitealizeComponent 语句做如下处理:

```
public Form1()
{
    InitializeComponent();
    Control.CheckForIllegalCrossThreadCalls = false;
}
```

第六步，异步写数据

```
private void Btn_Write_Click(object sender, EventArgs e)
{
        ……

        object[] values = new object[4];
        values[0] = Txt_W1.Text;
        values[1] = Txt_W2.Text;
        values[2] = "test";//采用常数
        values[3] = 1; //采用常数

        IOPCAsyncIO2Obj.Write(4, ItemServerHandle, values, 3, out nCancelid, out
                        pErrors); //异步写数据

        ……

}
```

写完成处理（执行结果监视）

```
public virtual void OnWriteComplete ( System.Int32 dwTransid ,
    System.Int32 hGroup ,
    System.Int32 hrMastererr ,
    System.Int32 dwCount ,
    int[] pClienthandles ,
    int[] pErrors )
{
        ……

        ServerObj.GetErrorString( pErrors[0], LOCALE_ID, out strResult);
        Txt_WriteStatus1.Text = strResult;

        ……

}
```

第七步，订阅方式读数据

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。工程应用中，大量数据的操作使用订阅方式更有优势。

订阅方式下，要考虑数据更新速度，及是否采用订阅方式读写。

```
private void CHK_Btn_CheckedChanged(object sender, EventArgs e)
{
   ……
   GCHandle hActive = GCHandle.Alloc(nActive, GCHandleType.Pinned);
   if (CHK_Btn.Checked != true)
       hActive.Target = 0;
   else
      hActive.Target = 1;
   ……
   IOPCGroupStateMgtObj.SetState(pRequestedUpdateRate, out nRevUpdateRate,
                                 hActive.AddrOfPinnedObject(), pTimeBias, pDeadband,
                                 pLCID, hClientGroup); //为组设定特定信息
   ……
}
```

通过IOPCDataCallback的虚函数OnDataChange实现

```
public virtual void OnDataChange(Int32 dwTransid,
       Int32 hGroup,
       Int32 hrMasterquality,
       Int32 hrMastererror,
       Int32 dwCount,
       int[] phClientItems,
       object[] pvValues, //值
       short[] pwQualities, //质量码
       OpcRcw.Da.FILETIME[] pftTimeStamps, //时间戳
       int[] pErrors)
{
       ……
       if (phClientItems[nCount] == 1) //根据Item在客户端注册句柄查询
       {
           Txt_R1_Value.Text = Convert.ToString(pvValues[nCount]);
           Txt_R1_Quality.Text = GetQuality(pwQualities[nCount]);
           DateTime dt = ToDateTime(pftTimeStamps[nCount]);
           Txt_R1_TimeStamp.Text = dt.ToString();
       }
       ……
}
```

第八步，注销相应实例

```
   private void Btn_Disconn_Click(object sender, EventArgs e)
          {
                ……
```

}

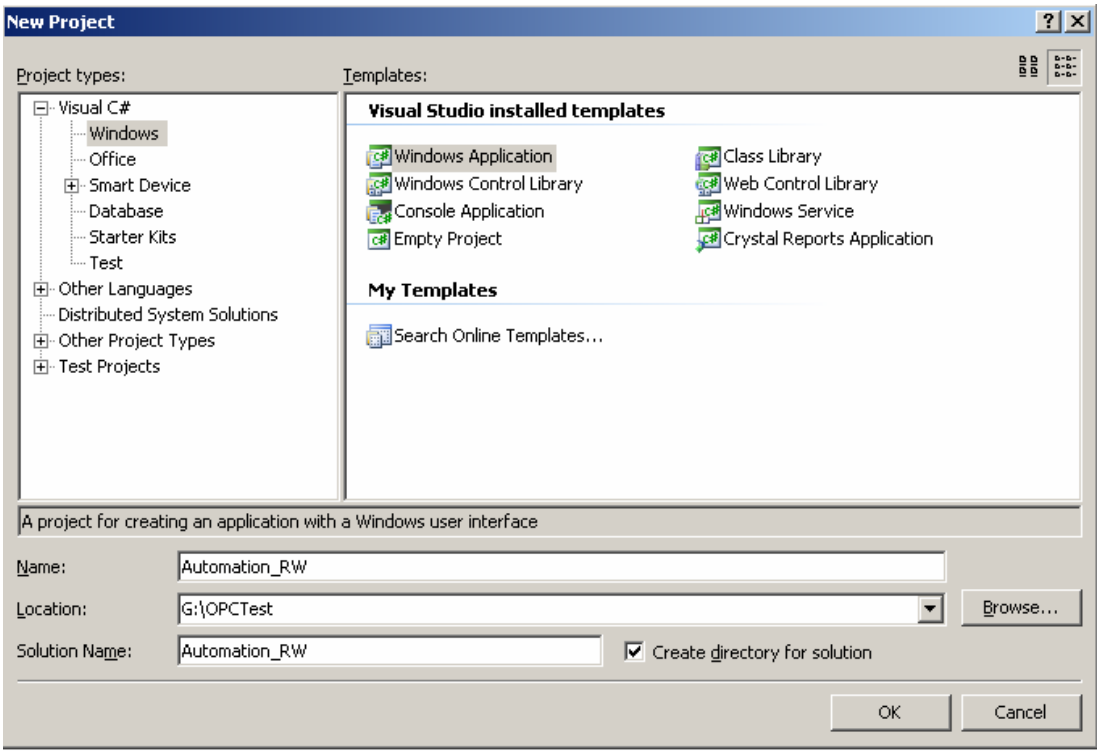
实例参考

**参考附录中 ASync_RW 例程。**

## 5、采用自动化接口实现过程

对于自动化接口，程序相应简单些。

建立项目：Automation_RW




测试中，对 db10.dbw0 及 db10.dbw2 读写操作，在 Form 窗口做如下设计：

| Control | name | Text |
|---|---|---|
| Button: | Btn_Conn | Conn |
| Button: | Btn_Read_S | Read_S |
| Button: | Btn_Read_A | Read_A |
| Button: | Btn_Write_S | Write_S |
| Button: | Btn_Write_A | Write_A |
| Button: | Btn_DisConn | disConn |

TextBox: Txt_R1_Value

TextBox: Txt_R1_Quality

TextBox: Txt_R1_TimeStamp

TextBox: Txt_R2_Value

TextBox: Txt_R2_Quality

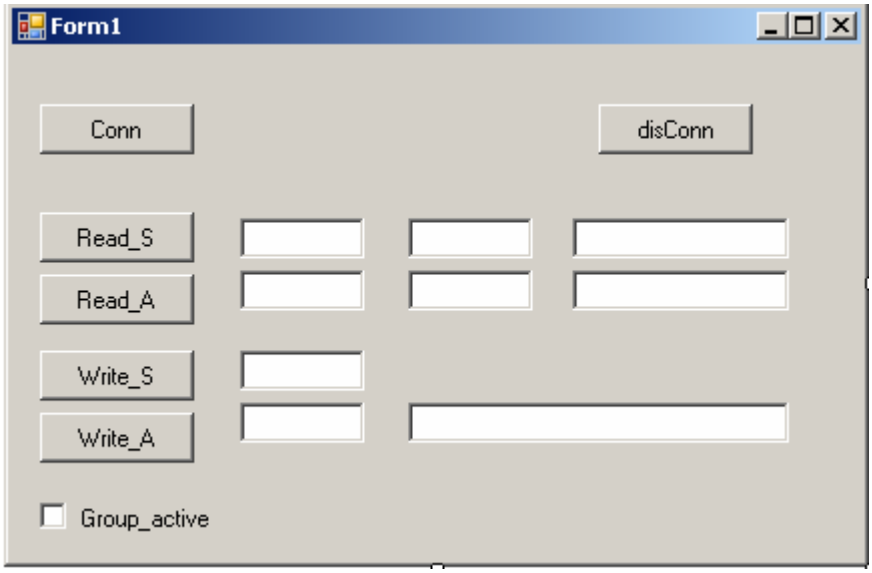
TextBox: Txt_R2_TimeStamp

TextBox: Txt_W1

TextBox: Txt_W2

TextBox: Txt_ Txt_WriteStatus2

CheckBox: CHK_Btn




第一步，添加下列命名空间（首先在 COM 组件中添加相应组件）

```
using OPCSiemensDAAutomation;
```

第二步，定义 OPC 相关变量

```
OPCServer MyOpcServer; //定义OPCServer
OPCGroup MyOpcGroup; //定义组
OPCItem MyOpcItem1; //Item
OPCItem MyOpcItem2; //值
long[] ServerHandle = new long[2]; //Item 的句柄
```

第三步，建立连接及对象

```
MyOpcServer = new OPCServer();
MyOpcServer.Connect("OPC.SimaticNet", "192.168.0.102");
MyOpcGroup = MyOpcServer.OPCGroups.Add("MyGroup1");

MyOpcItem1 = MyOpcGroup.OPCItems.AddItem("S7:[S7 connection_1]DB10,INT0",1);
MyOpcItem2 = MyOpcGroup.OPCItems.AddItem("S7:[S7 connection_1]DB10,INT2", 2);
ServerHandle[0] = MyOpcItem1.ServerHandle;
 ServerHandle[1] = MyOpcItem2.ServerHandle;
```

第四步，同步读数据，

```
private void Btn_Read_S_Click(object sender, EventArgs e)//同步读数据
{
        ……
        MyOpcItem1.Read(1,out ItemValues, out Qualities, out TimeStamps);
                        //ItemValues，Qualities，TimeStamps分别是值，质量码及时间
                        //也可以通过调用SyncRead函数，参数可参考异步读函数
        ……
}
```

第四步，同步写数据

```
private void Btn_Write_S_Click(object sender, EventArgs e)
{
        ……
        MyOpcItem1.Write(Txt_W1.Text);
                        //也可以通过调用SyncWrite函数，参数可参考异步写函数
        ……
}
```

第五步，异步事件定义，

在异步操作情况下，需要定义相应的异步事件

```
MyOpcGroup.DataChange += new
     DIOPCGroupEvent_DataChangeEventHandler(MyOpcGroup_DataChange); //
                                        //订阅方式下数据改变
iteComplete += new
     DIOPCGroupEvent_AsyncWriteCompleteEventHandler(MyOpcGroup_WriteComplete);
                                        //写完成事件
MyOpcGroup.AsyncReadComplete += new
     DIOPCGroupEvent_AsyncReadCompleteEventHandler(MyOpcGroup_ReadComplete);
                                        //读完成事件
MyOpcGroup.AsyncCancelComplete += new
     DIOPCGroupEvent_AsyncCancelCompleteEventHandler(MyOpcGroup_CancelComplete);
                                        //取消操作事件
```

在使用中注意，其事件函数要按照特定接口：

```
void MyOpcGroup_DataChange(int TransactionID, int NumItems, ref Array ClientHandles,
            ref Array ItemValues, ref Array Qualities, ref Array TimeStamps)
void MyOpcGroup_WriteComplete(int TransactionID, int NumItems, ref Array ClientHandles,
            ref Array Errors)
void MyOpcGroup_ReadComplete(int TransactionID, int NumItems, ref System.Array
            ClientHandles, ref System.Array ItemValues, ref System.Array Qualities,
            ref System.Array TimeStamps, ref System.Array Errors)
void MyOpcGroup_CancelComplete(int CancelID)
```

第六步，订阅方式读

```
void MyOpcGroup_DataChange(int TransactionID, int NumItems, ref Array ClientHandles,
      ref Array ItemValues, ref Array Qualities, ref Array TimeStamps)
{
        ……
        //注意数据改变时，Item 数量要通过 NumItems 得到，也就是说只有数据改变时，才对一
        遍，所以降低了服务器负担。要注意读语句写法。
```

```
            ……
        }
```

第七步，异步读

```
        private void Btn_Read_A_Click(object sender, EventArgs e)//异步读事件
        {
            int[] handle = new int[3] {ServerHandle[0], ServerHandle[1],0};//注意方式
            Array MyServerHandles = (Array)handle;
            Array errors;
            int cancelID;
                ……
            MyOpcGroup.AsyncRead(2, ref MyServerHandles, out errors, READASYNC__ID, out
                            cancelID);
                ……
        }
        void MyOpcGroup_ReadComplete(int TransactionID, int NumItems, ref System.Array
                    ClientHandles, ref System.Array ItemValues, ref System.Array Qualities,
                    ref System.Array TimeStamps, ref System.Array Errors)
        {
                ……
                //注意TransactionID的对应
                ……
        }
```

注意 array 在函数内部做参数时，数据下标是从 1 开始的，所以要考虑将第 0 位空出来，n 个 Item，就要定义 n+1 列数组，添加一个 0，但在函数使用时，又是从左开始读的。否则会报错。

第八步，异步写

```
        private void Btn_Write_A_Click(object sender, EventArgs e)
        {
                ……
                MyOpcGroup.AsyncWrite(2, ref MyServerHandles, ref Myvalues, out errors,
                       WRITEASYNC_ID, out cancelID);
                ……
        }
        void MyOpcGroup_WriteComplete(int TransactionID, int NumItems, ref Array ClientHandles,
                    ref Array Errors)
        {

                ……
        }
```

同样要注意 Array 在函数内部做参数的传递。

第九步，释放对象

```
    private void Btn_Disconn_Click(object sender, EventArgs e)
        {
                ……
        }
```

**参考附录中 Automation_RW 例程。**

## 6、OPCItem 的数据类型

在通过自定义接口访问时，

```
ItemArray[1].szAccessPath = "";
ItemArray[1].szItemID = "S7:[S7 connection_1]DB10,Real4";//地址，不同数据类型表示
ItemArray[1].bActive = 1;//是否激活
ItemArray[1].hClient = 2;//表示ID
ItemArray[1].dwBlobSize = 0;
ItemArray[1].pBlob = IntPtr.Zero;
ItemArray[1].vtRequestedDataType = 5;

ItemArray[2].szAccessPath = "";
ItemArray[2].szItemID = "S7:[S7 connection_1]DB10,STRING26.10";//地址，不同数据类型表
                                                    示方法不同
ItemArray[2].bActive = 1;//是否激活
ItemArray[2].hClient = 3;//表示ID
ItemArray[2].dwBlobSize = 0;
ItemArray[2].pBlob = IntPtr.Zero;
ItemArray[2].vtRequestedDataType = 8;
```

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。

| VbBoolean | VbByte | VbDecimal | VbDouble | Vbinteger | VbLong | VbSingle | VbString |
|---|---|---|---|---|---|---|---|
| 11 | 17 | 14 | 5 | 2 | 3 | 4 | 8 |

## 7、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"。

同时需要注意的是，测试环境客户端需要安装 simaticNet。

## 8、代码

### 8.1 自动化接口

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Collections;

using OPCSiemensDAAutomation;//引用连接库

namespace Automation_RW
{
    public partial class Form1 : Form
```

```
{
    public Form1()
    {
        InitializeComponent();
    }

    OPCServer MyOpcServer;//OPCServer
    OPCGroup MyOpcGroup;//
    OPCItem MyOpcItem1;
    OPCItem MyOpcItem2;
    int[] ServerHandle = new int[2];//服务器端注册句柄

    const int READASYNC__ID=1;//异步读事务
    const int WRITEASYNC_ID=2;//异步写事务



    private void Btn_Conn_Click(object sender, EventArgs e)//建立连接
    {
        try
        {
            MyOpcServer = new OPCServer();
            MyOpcServer.Connect("OPC.SimaticNet", "192.168.0.102");//OPCServer
            MyOpcGroup = MyOpcServer.OPCGroups.Add("MyGroup1");
            MyOpcGroup.IsActive = true;
            MyOpcGroup.IsSubscribed = true;//是否异步，在采用异步读写，订阅等方式下都需要
为
            MyOpcGroup.DeadBand = 0;
            MyOpcGroup.UpdateRate = 1000;//更新速率s

            MyOpcItem1 = MyOpcGroup.OPCItems.AddItem("S7:[S7 connection_1]DB10,INT0", 1);
            MyOpcItem2 = MyOpcGroup.OPCItems.AddItem("S7:[S7 connection_1]DB10,INT2", 2);
            ServerHandle[0] = MyOpcItem1.ServerHandle;
            ServerHandle[1] = MyOpcItem2.ServerHandle;

            MyOpcGroup.AsyncWriteComplete += new
                DIOPCGroupEvent_AsyncWriteCompleteEventHandler(MyOpcGroup_WriteComplete);
            MyOpcGroup.AsyncReadComplete += new
                DIOPCGroupEvent_AsyncReadCompleteEventHandler(MyOpcGroup_ReadComplete);
            MyOpcGroup.AsyncCancelComplete += new
                DIOPCGroupEvent_AsyncCancelCompleteEventHandler(MyOpcGroup_CancelComplete);
            MyOpcGroup.DataChange += new
                DIOPCGroupEvent_DataChangeEventHandler(MyOpcGroup_DataChange);

        }
        catch(System.Exception error)
        {
            MessageBox.Show(error.Message, "Result - connect server", MessageBoxButtons.OK,
                            MessageBoxIcon.Error);
        }

    }

    private void Btn_Read_S_Click(object sender, EventArgs e)//同步读数据
```

```
{

    object ItemValues;
    object Qualities;
    object TimeStamps;
    try
    {
        MyOpcItem1.Read(1,out ItemValues, out Qualities, out TimeStamps);
        Txt_R1_Value.Text = String.Format("{0}", ItemValues);
        // Quality
        Txt_R1_Quality.Text = String.Format("{0}", Qualities);
        // Timestamp
        Txt_R1_TimeStamp.Text = String.Format("{0}", TimeStamps);

    }
    catch (System.Exception error)
    {
        MessageBox.Show(error.Message, "Result - 同步读", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
    }
}
private void Btn_Write_S_Click(object sender, EventArgs e)//同步写数据
{
    try
    {
        MyOpcItem1.Write(Txt_W1.Text);
    }
    catch (System.Exception error)
    {
        MessageBox.Show(error.Message, "Result - 同步写", MessageBoxButtons.OK,
            MessageBoxIcon.Error);

    }
}


void MyOpcGroup_CancelComplete(int CancelID)
{
    //增加相应代码
}

private void Btn_Read_A_Click(object sender, EventArgs e)//异步读事件
{
    int[] handle = new int[3] {ServerHandle[0], ServerHandle[1],0};//注意写的方式
    Array MyServerHandles = (Array)handle;
    Array errors;
    int cancelID;
    try
    {
        MyOpcGroup.AsyncRead(2, ref MyServerHandles, out errors, READASYNC__ID, out
                    cancelID);

    }
    catch (System.Exception error)
    {
```

```
            MessageBox.Show(error.Message, "Result - 异步读", MessageBoxButtons.OK,
                MessageBoxIcon.Error);

        }
    }
    //读完成事件
    void MyOpcGroup_ReadComplete(int TransactionID, int NumItems, ref System.Array
            ClientHandles, ref System.Array ItemValues, ref System.Array Qualities,
            ref System.Array TimeStamps, ref System.Array Errors)
    {
        try
        {
            if (TransactionID == READASYNC__ID)
            {
                if(Convert.ToInt32(ClientHandles.GetValue(1))==1)
                {
                    if (Convert.ToInt32(Errors.GetValue(1))==0)
                    {
                        Txt_R2_Value.Text = ItemValues.GetValue(1).ToString();
                        Txt_R2_Quality.Text = Qualities.GetValue(1).ToString();
                        Txt_R2_TimeStamp.Text = TimeStamps.GetValue(1).ToString();
                    }
                }


            }
            //增加其余的代码
        }
        catch (System.Exception error)
        {
            MessageBox.Show(error.Message, "Result - 异步读", MessageBoxButtons.OK,
                    MessageBoxIcon.Error);

        }
    }

    //订阅方式
    void MyOpcGroup_DataChange(int TransactionID, int NumItems, ref Array ClientHandles,
                ref Array ItemValues, ref Array Qualities, ref Array TimeStamps)
    {
        try
        {
            for (int i = 0; i < NumItems; i++)
            {
                for (int j = 1; j < 3; j++)
                {
                    if (Convert.ToInt32(ClientHandles.GetValue(i + 1)) == j)
                    {
                        if (ItemValues.GetValue(i + 1) != null)
                        {

                            Txt_R2_Value.Text = ItemValues.GetValue(i + 1).ToString();
                            Txt_R2_Quality.Text = Qualities.GetValue(i + 1).ToString();
                            Txt_R2_TimeStamp.Text = TimeStamps.GetValue(i + 1).ToString();
                        }
```

```
                }
            }
        }
    }
    catch (System.Exception error)
    {
        MessageBox.Show(error.Message, "Result - 订阅", MessageBoxButtons.OK,
                MessageBoxIcon.Error);

    }
}

private void Btn_Write_A_Click(object sender, EventArgs e)//异步写
{
    int[] handle = new int[3] { ServerHandle[0], ServerHandle[1],0 };
    Array MyServerHandles = (Array)handle;
    object[] values = new object[3] {14,Txt_W2.Text,""};
    Array Myvalues=(Array)values;
    Array errors;
    int cancelID;
    try
    {
        MyOpcGroup.AsyncWrite(2, ref MyServerHandles, ref Myvalues, out errors,
                WRITEASYNC_ID, out cancelID);
    }
    catch (System.Exception error)
    {
        MessageBox.Show(error.Message, "Result - 异步写", MessageBoxButtons.OK,
                MessageBoxIcon.Error);

    }
}
//异步写完成
void MyOpcGroup_WriteComplete(int TransactionID, int NumItems, ref Array ClientHandles,
        ref Array Errors)
{
    Txt_WriteStatus2.Text = Errors.GetValue(1).ToString();
}

private void CHK_Btn_CheckedChanged(object sender, EventArgs e)
{
    if (CHK_Btn.Checked != true)
    {
        MyOpcGroup.IsSubscribed = false;
    }
    else
    {
        MyOpcGroup.IsSubscribed = true;
    }
}
//推出释放连接及对象
private void Btn_Disconn_Click(object sender, EventArgs e)
{
    try
```

```
            {
                if (MyOpcItem1 != null)
                    MyOpcItem1 = null;
                if (MyOpcItem2 != null)
                    MyOpcItem2 = null;
                if (MyOpcGroup != null)
                    MyOpcGroup = null;
                MyOpcServer.Disconnect();
            }
            catch (System.Exception error)
            {
                MessageBox.Show(error.Message, "Result - 异步写", MessageBoxButtons.OK,
                            MessageBoxIcon.Error);

            }
        }
    }
}
```

## 8.2 自定义接口同步读写

```
using System;
using System.Collections;
using System.Runtime.InteropServices;
using System.Drawing;
using System.ComponentModel;
using System.Windows.Forms;
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Data;

using OpcRcw.Comn;//引用
using OpcRcw.Da;//引用

namespace Sync_RW
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        OpcRcw.Da.IOPCServer ServerObj;//定义OPCServer 对象
        OpcRcw.Da.IOPCSyncIO IOPCSyncIO2Obj = null;//同步读对象
        OpcRcw.Da.IOPCGroupStateMgt IOPCGroupStateMgtObj = null;//管理OPCGroup组对象

        internal const int LOCALE_ID = 0x407;//OPCServer语言码-英语

        Object MyobjGroup1 = null;//OPCGroup对象
        int[] ItemServerHandle;//Item句柄数组
        int pSvrGroupHandle = 0;//OPCGroup 句柄

        private void Btn_Conn_Click(object sender, EventArgs e)
        {
```

```
Type svrComponenttyp;
Int32 dwRequestedUpdateRate = 1000;//订阅读取速度
Int32 hClientGroup = 1;
Int32 pRevUpdateRate;
OpcRcw.Da.OPCITEMDEF[] ItemArray;

float deadband = 0;

int TimeBias = 0;

GCHandle hTimeBias, hDeadband;
hTimeBias = GCHandle.Alloc(TimeBias, GCHandleType.Pinned);
hDeadband = GCHandle.Alloc(deadband, GCHandleType.Pinned);
Guid iidRequiredInterface = typeof(IOPCItemMgt).GUID;
try
{
    svrComponenttyp = Type.GetTypeFromProgID("OPC.SimaticNet",
                                "192.168.0.102");//OPCServer
    ServerObj = (OpcRcw.Da.IOPCServer)Activator.CreateInstance(svrComponenttyp);
                                                          //注册
    try
    {
        ServerObj.AddGroup("MyOPCGroup1",//增加组
            0,
            dwRequestedUpdateRate,
            hClientGroup,
            hTimeBias.AddrOfPinnedObject(),
            hDeadband.AddrOfPinnedObject(),
            LOCALE_ID,
            out pSvrGroupHandle,
            out pRevUpdateRate,
            ref iidRequiredInterface,
            out MyobjGroup1);

        IOPCSyncIO2Obj = (IOPCSyncIO)MyobjGroup1;
                          //Query interface for sync calls on group object
        IOPCGroupStateMgtObj = (IOPCGroupStateMgt)MyobjGroup1;
        ItemArray = new OPCITEMDEF[2];//定义读写的item，共个变量

        ItemArray[0].szAccessPath = "";
        ItemArray[0].szItemID = "S7:[S7 connection_1]DB10,INT0";
                         //地址，不同数据类型表示方法不同
        ItemArray[0].bActive = 1;//是否激活
        ItemArray[0].hClient = 1;//表示ID
        ItemArray[0].dwBlobSize = 0;
        ItemArray[0].pBlob = IntPtr.Zero;
        ItemArray[0].vtRequestedDataType = 2;

        ItemArray[1].szAccessPath = "";
        ItemArray[1].szItemID = "S7:[S7 connection_1]DB10,STRING14.10";
                         //地址，不同数据类型表示方法不同
        ItemArray[1].bActive = 1;//是否激活
        ItemArray[1].hClient = 2;//表示ID
        ItemArray[1].dwBlobSize = 0;
        ItemArray[1].pBlob = IntPtr.Zero;
```

```
        ItemArray[1].vtRequestedDataType =8;


        IntPtr pResults = IntPtr.Zero;
        IntPtr pErrors = IntPtr.Zero;
        try
        {
            ((OpcRcw.Da.IOPCItemMgt)MyobjGroup1).AddItems(2, ItemArray, out
                                pResults, out pErrors);

            int[] errors = new int[2];
            IntPtr pos = pResults;

            ItemServerHandle = new int[2];
            Marshal.Copy(pErrors, errors, 0, 2);
            if (errors[0] == 0)
            {
                OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                                typeof(OPCITEMRESULT));
                ItemServerHandle[0] = result.hServer;
            }
            if (errors[1] == 0)
            {
                pos = new IntPtr(pos.ToInt32() +
                                Marshal.SizeOf(typeof(OPCITEMRESULT)));
                OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                                typeof(OPCITEMRESULT));
                ItemServerHandle[1] = result.hServer;
            }
        }
        catch (System.Exception error) // catch for add items
        {
            MessageBox.Show(error.Message, "Result - Adding Items",
                                MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        finally
        {
            // Free the memory
            if (pResults != IntPtr.Zero)
            {
                Marshal.FreeCoTaskMem(pResults);
                pResults = IntPtr.Zero;
            }
            if (pErrors != IntPtr.Zero)
            {
                Marshal.FreeCoTaskMem(pErrors);
                pErrors = IntPtr.Zero;
            }
        }
    }
    catch (System.Exception error) // catch for group adding
    {
        MessageBox.Show(String.Format("Error while creating group object:-{0}",
                        error.Message), "Result - Add group", MessageBoxButtons.OK,
                        MessageBoxIcon.Error);
```

```
            }
            finally
            {
                if (hDeadband.IsAllocated) hDeadband.Free();
                if (hTimeBias.IsAllocated) hTimeBias.Free();
            }
        }
        catch (System.Exception error) // catch for server instance creation
        {
            MessageBox.Show(String.Format("Error while creating server object:-{0}",
                            error.Message), "Result - Create Server",
                            MessageBoxButtons.OK, MessageBoxIcon.Error);
        }

    }

    private void Btn_Read_Click(object sender, EventArgs e)//同步读
    {
        IntPtr pItemValues = IntPtr.Zero;
        IntPtr pErrors = IntPtr.Zero;
        try
        {
            IOPCSyncIO2Obj.Read(OPCDATASOURCE.OPC_DS_DEVICE, 2, ItemServerHandle, out
                            pItemValues, out pErrors);
            int[] errors = new int[2];
            Marshal.Copy(pErrors, errors, 0, 2);
            OPCITEMSTATE[] pItemState = new OPCITEMSTATE[2];
            if (errors[0] == 0)
            {
                pItemState[0] = (OPCITEMSTATE)Marshal.PtrToStructure(pItemValues,
                            typeof(OPCITEMSTATE));
                pItemValues=new
                            IntPtr(pItemValues.ToInt32()+Marshal.SizeOf(typeof(OPCITEMS
                            TATE)));
                // update the UI
                //txt_R1.Text = String.Format("{0}",pItemState.vDataValue);
                Txt_R1_Value.Text = String.Format("{0}", pItemState[0].vDataValue);
                Txt_R1_Quality.Text = GetQuality(pItemState[0].wQuality);
                DateTime dt = ToDateTime(pItemState[0].ftTimeStamp);
                Txt_R1_TimeStamp.Text = dt.ToString();
                // quality
            }
            if (errors[1] == 0)
            {
                pItemState[1]= (OPCITEMSTATE)Marshal.PtrToStructure(pItemValues,
                            typeof(OPCITEMSTATE));
                pItemValues = new IntPtr(pItemValues.ToInt32()+
                            Marshal.SizeOf(typeof(OPCITEMSTATE)));
                // update the UI
                Txt_R2_Value.Text = String.Format("{0}", pItemState[1].vDataValue);
                Txt_R2_Quality.Text = GetQuality(pItemState[1].wQuality);
                DateTime dt = ToDateTime(pItemState[1].ftTimeStamp);
                Txt_R2_TimeStamp.Text = dt.ToString();
                // quality
            }
```

```
        }
        catch (System.Exception error)
        {
            MessageBox.Show(error.Message, "Result - Read Items", MessageBoxButtons.OK,
                            MessageBoxIcon.Error);
        }
        finally
        {
            // Free the unmanaged memory
            if (pItemValues != IntPtr.Zero)
            {
                Marshal.FreeCoTaskMem(pItemValues);
                pItemValues = IntPtr.Zero;
            }
            if (pErrors != IntPtr.Zero)
            {
                Marshal.FreeCoTaskMem(pErrors);
                pErrors = IntPtr.Zero;
            }
        }
    }

    private void Btn_Write_Click(object sender, EventArgs e)//同步写
    {
        IntPtr pErrors = IntPtr.Zero;

        object[] values = new object[2];
        values[0] = Txt_W1.Text;
        values[1] = Txt_W2.Text;
        try
        {
            IOPCSyncIO2Obj.Write(2, ItemServerHandle, values, out pErrors);
            int[] errors = new int[2];
            Marshal.Copy(pErrors, errors, 0, 2);

            String pstrError;
            String pstrError1;
            ServerObj.GetErrorString(errors[0], LOCALE_ID, out pstrError);
            ServerObj.GetErrorString(errors[1], LOCALE_ID, out pstrError1);

        }
        catch (System.Exception error)
        {
            MessageBox.Show(error.Message, "Result - WriteItem", MessageBoxButtons.OK,
                            MessageBoxIcon.Error);
        }
        finally
        {
            if (pErrors != IntPtr.Zero)
            {
                Marshal.FreeCoTaskMem(pErrors);
                pErrors = IntPtr.Zero;
            }
        }
```

```
}

private String GetQuality(long wQuality)//质量码
{
    String strQuality = "";
    switch (wQuality)
    {
        case Qualities.OPC_QUALITY_GOOD:
            strQuality = "Good";
            break;
        case Qualities.OPC_QUALITY_BAD:
            strQuality = "Bad";
            break;
        case Qualities.OPC_QUALITY_CONFIG_ERROR:
            strQuality = "BadConfigurationError";
            break;
        case Qualities.OPC_QUALITY_NOT_CONNECTED:
            strQuality = "BadNotConnected";
            break;
        case Qualities.OPC_QUALITY_DEVICE_FAILURE:
            strQuality = "BadDeviceFailure";
            break;
        case Qualities.OPC_QUALITY_SENSOR_FAILURE:
            strQuality = "BadSensorFailure";
            break;
        case Qualities.OPC_QUALITY_COMM_FAILURE:
            strQuality = "BadCommFailure";
            break;
        case Qualities.OPC_QUALITY_OUT_OF_SERVICE:
            strQuality = "BadOutOfService";
            break;
        case Qualities.OPC_QUALITY_WAITING_FOR_INITIAL_DATA:
            strQuality = "BadWaitingForInitialData";
            break;
        case Qualities.OPC_QUALITY_EGU_EXCEEDED:
            strQuality = "UncertainEGUExceeded";
            break;
        case Qualities.OPC_QUALITY_SUB_NORMAL:
            strQuality = "UncertainSubNormal";
            break;
        default:
            strQuality = "Not handled";
            break;
    }

    return strQuality;
}
private DateTime ToDateTime(OpcRcw.Da.FILETIME ft)
{
    long highbuf = (long)ft.dwHighDateTime;
    long buffer = (highbuf << 32) + ft.dwLowDateTime;
    return DateTime.FromFileTimeUtc(buffer);
}

private void Btn_Disconn_Click(object sender, EventArgs e)//对象注销，断开连接
```

```
        {
            try
            {

                if (IOPCSyncIO2Obj != null)
                {
                    Marshal.ReleaseComObject(IOPCSyncIO2Obj);
                    IOPCSyncIO2Obj = null;
                }
                ServerObj.RemoveGroup(pSvrGroupHandle, 0);
                if (IOPCGroupStateMgtObj != null)
                {
                    Marshal.ReleaseComObject(IOPCGroupStateMgtObj);
                    IOPCGroupStateMgtObj = null;
                }
                if (MyobjGroup1 != null)
                {
                    Marshal.ReleaseComObject(MyobjGroup1);
                    MyobjGroup1 = null;
                }
                if (ServerObj != null)
                {
                    Marshal.ReleaseComObject(ServerObj);
                    ServerObj = null;
                }
            }
            catch (System.Exception error)
            {
                MessageBox.Show(error.Message, "Result - Stop Server", MessageBoxButtons.OK,
                          MessageBoxIcon.Error);
            }
        }
    }
}
```

## 8.3 自定义接口异步读写

```
using System;
using System.Collections;
using System.Runtime.InteropServices;
using System.Drawing;
using System.ComponentModel;
using System.Windows.Forms;
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Data;

using OpcRcw.Comn;//引用
using OpcRcw.Da;//引用

namespace ASync_RW
{
    public partial class Form1 : Form, IOPCDataCallback
    {
```

```
public Form1()
{
    InitializeComponent();
    Control.CheckForIllegalCrossThreadCalls = false;
}
OpcRcw.Da.IOPCServer ServerObj;//OPCServer
OpcRcw.Da.IOPCAsyncIO2 IOPCAsyncIO2Obj = null;//异步读写对象
OpcRcw.Da.IOPCGroupStateMgt IOPCGroupStateMgtObj = null;//组管理对象

IConnectionPointContainer pIConnectionPointContainer = null;
IConnectionPoint pIConnectionPoint = null;

internal const int LOCALE_ID = 0x407;

Object MyobjGroup1 = null;
int[] ItemServerHandle;
int pSvrGroupHandle = 0;
Int32 dwCookie = 0;
private void Btn_Conn_Click(object sender, EventArgs e)
{
    Type svrComponenttyp;
    Int32 dwRequestedUpdateRate = 1000;
    Int32 hClientGroup = 1;
    Int32 pRevUpdateRate;
    OpcRcw.Da.OPCITEMDEF[] ItemArray;

    float deadband = 0;

    int TimeBias = 0;

    GCHandle hTimeBias, hDeadband;
    hTimeBias = GCHandle.Alloc(TimeBias, GCHandleType.Pinned);
    hDeadband = GCHandle.Alloc(deadband, GCHandleType.Pinned);
    Guid iidRequiredInterface = typeof(IOPCItemMgt).GUID;
    try
    {
        svrComponenttyp = Type.GetTypeFromProgID("OPC.SimaticNet",
                 "192.168.0.102");//OPCServer
        ServerObj = (OpcRcw.Da.IOPCServer)Activator.CreateInstance(svrComponenttyp);
                                 //注册
        try
        {
            ServerObj.AddGroup("MyOPCGroup1",//组对象
                0,
                dwRequestedUpdateRate,
                hClientGroup,
                hTimeBias.AddrOfPinnedObject(),
                hDeadband.AddrOfPinnedObject(),
                LOCALE_ID,
                out pSvrGroupHandle,
                out pRevUpdateRate,
                ref iidRequiredInterface,
                out MyobjGroup1);

            IOPCAsyncIO2Obj = (IOPCAsyncIO2)MyobjGroup1;
```

```
                //Query interface for Async calls on group object

IOPCGroupStateMgtObj = (IOPCGroupStateMgt)MyobjGroup1;

pIConnectionPointContainer = (IConnectionPointContainer)MyobjGroup1;
                //定义特定组的异步调用连接

Guid iid = typeof(IOPCDataCallback).GUID;
                // Establish Callback for all async operations
pIConnectionPointContainer.FindConnectionPoint(ref iid, out
                pIConnectionPoint);

// Creates a connection between the OPC servers's connection point and
                this client's sink (the callback object)
pIConnectionPoint.Advise(this, out dwCookie);

ItemArray = new OPCITEMDEF[4];//定义读写的item，共个变量

ItemArray[0].szAccessPath = "";
ItemArray[0].szItemID = "S7:[S7 connection_1]DB10,INT0";
                //地址，不同数据类型表示方法不同
ItemArray[0].bActive = 1;//是否激活
ItemArray[0].hClient = 1;//表示ID
ItemArray[0].dwBlobSize = 0;
ItemArray[0].pBlob = IntPtr.Zero;
ItemArray[0].vtRequestedDataType = 2;

ItemArray[1].szAccessPath = "";
ItemArray[1].szItemID = "S7:[S7 connection_1]DB10,Real4";
                //地址，不同数据类型表示方法不同
ItemArray[1].bActive = 1;//是否激活
ItemArray[1].hClient = 2;//表示ID
ItemArray[1].dwBlobSize = 0;
ItemArray[1].pBlob = IntPtr.Zero;
ItemArray[1].vtRequestedDataType = 5;

ItemArray[2].szAccessPath = "";
ItemArray[2].szItemID = "S7:[S7 connection_1]DB10,STRING26.10";
                //地址，不同数据类型表示方法不同
ItemArray[2].bActive = 1;//是否激活
ItemArray[2].hClient = 3;//表示ID
ItemArray[2].dwBlobSize = 0;
ItemArray[2].pBlob = IntPtr.Zero;
ItemArray[2].vtRequestedDataType = 8;
IntPtr pResults = IntPtr.Zero;
IntPtr pErrors = IntPtr.Zero;

ItemArray[3].szAccessPath = "";
ItemArray[3].szItemID = "S7:[S7 connection_1]DB10,X12.0";
                //地址，不同数据类型表示方法不同
ItemArray[3].bActive = 1;//是否激活
ItemArray[3].hClient = 4;//表示ID
ItemArray[3].dwBlobSize = 0;
ItemArray[3].pBlob = IntPtr.Zero;
ItemArray[3].vtRequestedDataType = 11;
```

```
try
{
    ((OpcRcw.Da.IOPCItemMgt)MyobjGroup1).AddItems(4, ItemArray, out
                    pResults, out pErrors);

    int[] errors = new int[4];
    IntPtr pos = pResults;

    ItemServerHandle = new int[4];
    Marshal.Copy(pErrors, errors, 0, 4);
    if (errors[0] == 0)
    {
        OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                    typeof(OPCITEMRESULT));
        ItemServerHandle[0] = result.hServer;
    }
    if (errors[1] == 0)
    {
        pos = new IntPtr(pos.ToInt32() +
                    Marshal.SizeOf(typeof(OPCITEMRESULT)));
        OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                    typeof(OPCITEMRESULT));
        ItemServerHandle[1] = result.hServer;
    }
    if (errors[2] == 0)
    {
        pos = new IntPtr(pos.ToInt32() +
                    Marshal.SizeOf(typeof(OPCITEMRESULT)));
        OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                    typeof(OPCITEMRESULT));
        ItemServerHandle[2] = result.hServer;
    }
    if (errors[3] == 0)
    {
        pos = new IntPtr(pos.ToInt32() +
                    Marshal.SizeOf(typeof(OPCITEMRESULT)));
        OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pos,
                    typeof(OPCITEMRESULT));
        ItemServerHandle[3] = result.hServer;
    }
}
catch (System.Exception error) // catch for add items
{
    MessageBox.Show(error.Message, "Result - Adding Items",
                    MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
    // Free the memory
    if (pResults != IntPtr.Zero)
    {
        Marshal.FreeCoTaskMem(pResults);
        pResults = IntPtr.Zero;
    }
```

```
                    if (pErrors != IntPtr.Zero)
                    {
                        Marshal.FreeCoTaskMem(pErrors);
                        pErrors = IntPtr.Zero;
                    }
                }
            }
            catch (System.Exception error) // catch for group adding
            {
                MessageBox.Show(String.Format("Error while creating group object:-{0}",
                        error.Message), "Result - Add group", MessageBoxButtons.OK,
                        MessageBoxIcon.Error);
            }
            finally
            {
                if (hDeadband.IsAllocated) hDeadband.Free();
                if (hTimeBias.IsAllocated) hTimeBias.Free();
            }
        }
        catch (System.Exception error) // catch for server instance creation
        {
            MessageBox.Show(String.Format("Error while creating server object:-{0}",
                        error.Message), "Result - Create Server", MessageBoxButtons.OK,
                        MessageBoxIcon.Error);
        }

    }

    private void Btn_Read_Click(object sender, EventArgs e)//异步读
    {
        int nCancelid;
        IntPtr pErrors = IntPtr.Zero;
        if (IOPCAsyncIO2Obj != null)
        {
            try
            {
                IOPCAsyncIO2Obj.Read(4, ItemServerHandle, 2, out nCancelid, out pErrors);
                int[] errors = new int[4];
                Marshal.Copy(pErrors, errors, 0, 4);

            }
            catch (System.Exception error)
            {
               // MessageBox.Show(error.Message, "Error-Async Read", MessageBoxButtons.OK,
                                MessageBoxIcon.Error);
            }
        }
    }
    public virtual void OnReadComplete(System.Int32 dwTransid,//异步读完成
        System.Int32 hGroup,
        System.Int32 hrMasterquality,
        System.Int32 hrMastererror,
        System.Int32 dwCount,
        int[] phClientItems,
        object[] pvValues,//值
```

```
            short[] pwQualities,//质量码
            OpcRcw.Da.FILETIME[] pftTimeStamps,//事件戳
            int[] pErrors)
        {
            try
            {
                if (pErrors[0] == 0)
                {
                    string aa;
                    // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = false;
                    // Value
                    Txt_R1_Value.Text = String.Format("{0}", pvValues[0]);
                  //  txt_R4.Text = String.Format("{0}", pvValues[1]);


                    // Quality
                    Txt_R1_Quality.Text = GetQuality(pwQualities[0]);
                    // Timestamp
                    DateTime dt = ToDateTime(pftTimeStamps[0]);
                    Txt_R1_TimeStamp.Text = dt.ToString();
                    // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = true;
                }
                else
                {
                    String strResult = "";
                    ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strResult);
                    MessageBox.Show(strResult, "Result - OnReadCOmpleate",
                                    MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                if (pErrors[1] == 0)
                {
                    // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = false;
                    // Value
                    Txt_R2_Value.Text = String.Format("{0}", pvValues[1]);
                    //  txt_R4.Text = String.Format("{0}", pvValues[1]);


                    // Quality
                    Txt_R2_Quality.Text = GetQuality(pwQualities[1]);
                    // Timestamp
                    DateTime dt = ToDateTime(pftTimeStamps[1]);
                    Txt_R2_TimeStamp.Text = dt.ToString();
                    // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = true;
                }
                else
                {
                    String strResult = "";
                    ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strResult);
                    MessageBox.Show(strResult, "Result - OnReadCOmpleate",
                                    MessageBoxButtons.OK, MessageBoxIcon.Error);
                }

                if (pErrors[2] == 0)
                {
                    // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = false;
```

```
                // Value
                Txt_R3_Value.Text = String.Format("{0}", pvValues[2]);
                //  txt_R4.Text = String.Format("{0}", pvValues[1]);


                // Quality
                Txt_R3_Quality.Text = GetQuality(pwQualities[2]);
                // Timestamp
                DateTime dt = ToDateTime(pftTimeStamps[2]);
                Txt_R3_TimeStamp.Text = dt.ToString();
                // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = true;
            }
            else
            {
                String strResult = "";
                ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strResult);
                MessageBox.Show(strResult, "Result - OnReadCOmpleate",
                                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }

            if (pErrors[3] == 0)
            {
                // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = false;
                // Value
                Txt_R4_Value.Text = String.Format("{0}", pvValues[3]);
                //  txt_R4.Text = String.Format("{0}", pvValues[1]);


                // Quality
                Txt_R4_Quality.Text = GetQuality(pwQualities[3]);
                // Timestamp
                DateTime dt = ToDateTime(pftTimeStamps[3]);
                Txt_R4_TimeStamp.Text = dt.ToString();
                // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = true;
            }
            else
            {
                String strResult = "";
                ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strResult);
                MessageBox.Show(strResult, "Result - OnReadCOmpleate",
                                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        catch (System.Exception exp)
        {
            MessageBox.Show(exp.Message, "OnReadComplete-Runtime Error",
                                MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    public virtual void OnCancelComplete(System.Int32 dwTransid, System.Int32 hGroup)
    {
        // Not implemented in this sample.
    }
    public virtual void OnDataChange(Int32 dwTransid,//订阅方式
```

```
            Int32 hGroup,
            Int32 hrMasterquality,
            Int32 hrMastererror,
            Int32 dwCount,
            int[] phClientItems,
            object[] pvValues,
            short[] pwQualities,
            OpcRcw.Da.FILETIME[] pftTimeStamps,
            int[] pErrors)
        {
            try
            {

                for (int nCount = 0; nCount < dwCount; nCount++)
                {
                    if (pErrors[nCount] == 0)
                    {
                        if (phClientItems[nCount] == 1)
                        {
                            Txt_R1_Value.Text = Convert.ToString(pvValues[nCount]);
                            Txt_R1_Quality.Text = GetQuality(pwQualities[nCount]);
                            DateTime dt = ToDateTime(pftTimeStamps[nCount]);
                            Txt_R1_TimeStamp.Text = dt.ToString();
                        }
                        if (phClientItems[nCount] == 2)
                        {
                            Txt_R2_Value.Text = Convert.ToString(pvValues[nCount]);
                            Txt_R2_Quality.Text = GetQuality(pwQualities[nCount]);
                            DateTime dt = ToDateTime(pftTimeStamps[nCount]);
                            Txt_R2_TimeStamp.Text = dt.ToString();
                        }
                        if (phClientItems[nCount] == 3)
                        {
                            Txt_R3_Value.Text = Convert.ToString(pvValues[nCount]);
                            Txt_R3_Quality.Text = GetQuality(pwQualities[nCount]);
                            DateTime dt = ToDateTime(pftTimeStamps[nCount]);
                            Txt_R3_TimeStamp.Text = dt.ToString();
                        }
                        if (phClientItems[nCount] == 4)
                        {
                            Txt_R4_Value.Text = Convert.ToString(pvValues[nCount]);
                            Txt_R4_Quality.Text = GetQuality(pwQualities[nCount]);
                            DateTime dt = ToDateTime(pftTimeStamps[nCount]);
                            Txt_R4_TimeStamp.Text = dt.ToString();
                        }
                    }
                    else
                    {
                        String strItemErr;
                        ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strItemErr);
                        //MessageBox.Show(strItemErr, "OnDataChange-Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                }
            }
```

```
        catch (System.Exception exp)
        {
            MessageBox.Show(exp.Message, "OnDataChange-Runtime Error",
                            MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    private String GetQuality(long wQuality)
    {
        String strQuality = "";
        switch (wQuality)
        {
            case Qualities.OPC_QUALITY_GOOD:
                strQuality = "Good";
                break;
            case Qualities.OPC_QUALITY_BAD:
                strQuality = "Bad";
                break;
            case Qualities.OPC_QUALITY_CONFIG_ERROR:
                strQuality = "BadConfigurationError";
                break;
            case Qualities.OPC_QUALITY_NOT_CONNECTED:
                strQuality = "BadNotConnected";
                break;
            case Qualities.OPC_QUALITY_DEVICE_FAILURE:
                strQuality = "BadDeviceFailure";
                break;
            case Qualities.OPC_QUALITY_SENSOR_FAILURE:
                strQuality = "BadSensorFailure";
                break;
            case Qualities.OPC_QUALITY_COMM_FAILURE:
                strQuality = "BadCommFailure";
                break;
            case Qualities.OPC_QUALITY_OUT_OF_SERVICE:
                strQuality = "BadOutOfService";
                break;
            case Qualities.OPC_QUALITY_WAITING_FOR_INITIAL_DATA:
                strQuality = "BadWaitingForInitialData";
                break;
            case Qualities.OPC_QUALITY_EGU_EXCEEDED:
                strQuality = "UncertainEGUExceeded";
                break;
            case Qualities.OPC_QUALITY_SUB_NORMAL:
                strQuality = "UncertainSubNormal";
                break;
            default:
                strQuality = "Not handled";
                break;
        }

        return strQuality;
    }
    private DateTime ToDateTime(OpcRcw.Da.FILETIME ft)
    {
        long highbuf = (long)ft.dwHighDateTime;
        long buffer = (highbuf << 32) + ft.dwLowDateTime;
```

```
        return DateTime.FromFileTimeUtc(buffer);
    }

    private void Btn_Write_Click(object sender, EventArgs e)//
    {
        int nCancelid;
        IntPtr pErrors = IntPtr.Zero;
        object[] values = new object[4];
        values[0] = Txt_W1.Text;
        values[1] = Txt_W2.Text;
        values[2] = "test";
        values[3] = 1;
        if (IOPCAsyncIO2Obj != null)
        {
            try
            {
                IOPCAsyncIO2Obj.Write(4, ItemServerHandle, values, 3, out nCancelid, out
                            pErrors);
                int[] errors = new int[4];
                Marshal.Copy(pErrors, errors, 0, 4);
                if (errors[0] != 0 || errors[1] != 0)
                {
                    System.Exception ex = new Exception("Error in reading item");
                    Marshal.FreeCoTaskMem(pErrors);
                    pErrors = IntPtr.Zero;
                    throw ex;
                }
            }
            catch (System.Exception error)
            {
                MessageBox.Show(error.Message, "Result-Async Read", MessageBoxButtons.OK,
                            MessageBoxIcon.Error);
            }
        }
    }
    public virtual void OnWriteComplete(System.Int32 dwTransid,//写完成
       System.Int32 hGroup,
       System.Int32 hrMastererr,
       System.Int32 dwCount,
       int[] pClienthandles,
       int[] pErrors)
    {
        // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = false;
        // .Net 2.0 ThreadExceptionDialog.CheckForIllegalCrossThreadCalls = true;
        String strResult = "";
        String strResult1 = "";
        String strResult2 = "";
        String strResult3 = "";

        ServerObj.GetErrorString(pErrors[0], LOCALE_ID, out strResult);
        ServerObj.GetErrorString(pErrors[1], LOCALE_ID, out strResult1);
        ServerObj.GetErrorString(pErrors[2], LOCALE_ID, out strResult2);
        ServerObj.GetErrorString(pErrors[3], LOCALE_ID, out strResult3);
        Txt_WriteStatus1.Text = strResult;
        Txt_WriteStatus2.Text = strResult1;
```

```
}
private void CHK_Btn_CheckedChanged(object sender, EventArgs e)
{
    IntPtr pRequestedUpdateRate = IntPtr.Zero;
    int nRevUpdateRate = 0;
    IntPtr hClientGroup = IntPtr.Zero;
    IntPtr pTimeBias = IntPtr.Zero;
    IntPtr pDeadband = IntPtr.Zero;
    IntPtr pLCID = IntPtr.Zero;
    int nActive = 0;

    // activates or deactivates group according to checkbox status
    GCHandle hActive = GCHandle.Alloc(nActive, GCHandleType.Pinned);
    if (CHK_Btn.Checked != true)
        hActive.Target = 0;
    else
        hActive.Target = 1;
    try
    {
        IOPCGroupStateMgtObj.SetState(pRequestedUpdateRate, out nRevUpdateRate,
                  hActive.AddrOfPinnedObject(), pTimeBias, pDeadband, pLCID,
                  hClientGroup);
    }
    catch (System.Exception error)
    {
        MessageBox.Show(error.Message, "Result-Change Group State",
                  MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        hActive.Free();
    }
}

private void Btn_Disconn_Click(object sender, EventArgs e)//释放对象及断开连接
{
    try
    {

        CHK_Btn.Checked = false;
        if (dwCookie != 0)
        {
            pIConnectionPoint.Unadvise(dwCookie);
            dwCookie = 0;
        }
        // Free unmanaged code
        Marshal.ReleaseComObject(pIConnectionPoint);
        pIConnectionPoint = null;

        Marshal.ReleaseComObject(pIConnectionPointContainer);
        pIConnectionPointContainer = null;

        if (IOPCAsyncIO2Obj != null)
        {
```

```
                    Marshal.ReleaseComObject(IOPCAsyncIO2Obj);
                    IOPCAsyncIO2Obj = null;
                }

                ServerObj.RemoveGroup(pSvrGroupHandle, 0);
                if (IOPCGroupStateMgtObj != null)
                {
                    Marshal.ReleaseComObject(IOPCGroupStateMgtObj);
                    IOPCGroupStateMgtObj = null;
                }
                if (MyobjGroup1 != null)
                {
                    Marshal.ReleaseComObject(MyobjGroup1);
                    MyobjGroup1 = null;
                }
                if (ServerObj != null)
                {
                    Marshal.ReleaseComObject(ServerObj);
                    ServerObj = null;
                }
            }
            catch (System.Exception error)
            {
                MessageBox.Show(error.Message, "Result - Stop Server", MessageBoxButtons.OK,
                                MessageBoxIcon.Error);
            }
        }
    }
}
```

## 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。



**西门子（中国）有限公司**