OPC技术在工业监控系统中的应用

周晓斌

（海南红塔卷烟有限责任公司，海南 海口 571100）

摘 要：介绍了OPC技术的基础概念、技术规范及主要程序模型，针对目前采用的主流开发平台，介绍了OPC技术的应用开发方式。提供了3个不同类型的工程应用实例，通过不同平台下开发方式的选择，结合具体工程实例对OPC技术的不同开发方式进行比较，增进工程技术人员对OPC技术相关知识和应用开发方式的了解，为OPC技术的应用开发提供参考。

关键词：OPC技术；工业监控；应用开发

中图分类号：TP277.2 **文献标志码**：A **文章编号**：1673-9833(2011)05-0080-06

The Application of OPC in Industry Monitor System

Zhou Xiaobin

（Hainan Hongta Cigarette Co., Ltd., Haikou 571100, China）

Abstract: Expounds the basic concept, technical regulations and program modules of OPC, introduces the application and development approaches of OPC technology in view of currently used developing platform and provides three different types of engineering application. Through the choice of developing ways at different platforms and combination with specific project examples, compares the different developments of OPC technology, enhances the understanding of engineering and technical personnel to OPC technology-related knowledge and the applied developing methods, and provides a reference for the application and development of OPC technology.

Keywords: OPC technology; industry monitor; application development

0 引言

随着计算机技术和控制技术的不断发展，现代工业过程控制系统已逐渐发展为现场设备管理、过程管理和商业管理3个层次组成的综合系统。由于不同工业领域往往使用不同的专门设备、控制系统和应用程序，因此它们都面临着一个共同的问题，即如何在这些组件之间及与企业其它部门之间实现数据共享，解决它们之间的相互通信问题[1]。上述问题的关键是不同计算机系统（PCS，MES等）的接口不统一、不标准，过程控制系统和信息系统各有专用技术接口及应用程序接口（application program interface，API），尽管可以编写定制的驱动程序和接口程序，但因不同类型硬件及软件包都需相互通信，使得驱动程序的种类迅速增长，并且连接程序开发没有统一、开放的标准，不同程序间容易相互冲突，这种情况不仅增加了用户的负担，而且在实践中并不能真正解决不同系统的互操作性。出于对上述问题的考虑，1996年8月，一个由自动化领域的领先公司组成的工作组在Microsoft公司帮助下提出了一

收稿日期：2011-03-25

作者简介：周晓斌（1966-），男，湖南汨罗人，海南红塔卷烟有限责任公司工程师，主要研究方向为卷烟厂包装机电气控制，E-mail：zxbin@hongta.com

个基于微软 OLE，COM，DCOM 技术的开放的、灵活的、即插即用的工业标准 OPC（OLE for process control，用于过程控制的 OLE），不同厂商只要遵循 OPC 技术标准就可以实现软硬件的互操作性。

1　OPC 技术简介

OPC 是连接数据源（OPC 服务器）和数据的使用者（OPC 应用程序）之间的软件接口标准。数据源可以是 PLC，DCS 和条形码读取器等控制设备。随着控制系统构成的不同，作为数据源的 OPC 服务器既可以是和 OPC 应用程序在同一台计算机上运行的本地 OPC 服务器，也可以是在另外的计算机上运行的远程 OPC 服务器。OPC 接口既可用于通过网络把最下层控制设备的原始数据提供给数据使用者（OPC 应用程序、人机界面等自动化程序，甚至更上层的历史数据库等应用程序），也可用于应用程序和物理设备的直接连接。所以，OPC 是适用于不同系统的具有高度柔性的接口标准。

1.1　OPC服务器

从概念的角度来讲，一个 OPC 服务器可以分解为 3 个模块：OPC 通讯模块、翻译 / 映射模块和本地通讯模块。

OPC 通讯模块：主要负责与指定的 OPC 客户端进行通讯。

本地通讯模块：OPC 服务器需要使用最有效的与数据发送设备通讯的方法，这些一般由设备供应商提供详细的解决方案。

翻译 / 映射模块：这个模块的任务是翻译来自 OPC 客户端的 OPC 请求，并且将这些请求转换为要被送达的数据发送端的本地请求，反之亦然。如果有效地做到了这一点，OPC 供应商能够将数据发送端的负荷保持在最小，同时使数据吞吐量达到最大值[2]6-7。

1.2　OPC客服端

与 OPC 服务器类似，OPC 客户端也可以被当作由 3 个模块组成：本地应用程序通讯、翻译 / 映射模块及 OPC 通讯模块。

OPC 通讯模块：它与 OPC 服务器连接交换数据，且在不破坏 OPC 服务器稳定性的情况下断开连接。

翻译 / 映射模块：将数据从终端设备或数据发送端读出，或将数据写入终端设备或数据发送端时，进行信息翻译。

应用程序通讯模块：OPC客户端主要是在一个特定的应用程序内部运行的，因此，它需要对应用程序的编程接口（API）进行几次呼叫，以便让数据通过 OPC 客户端，从应用程序传递到 OPC 服务器，然后到数据发送端。还有一种可能，是让一个普通的 OPC 客户端通过一个通信协议而不是通过一个 API 来与一个应用程序通讯，如果应用程序支持这样的协议[2]8-9。

2　OPC 应用开发方式

OPC 开发可分为 OPC 服务器和 OPC 客户端 2 类。OPC 服务器一般由设备供应商开发，在出售其硬件产品时，设备供应商会把 OPC 服务器程序提供给用户，本文主要介绍 OPC 客户端（应用程序）的开发。

OPC 由 2 套接口（见图 1）组成：客户端和服务器程序员使用的 OPC 自定义接口（OPC COM custom interfaces），支持高端商业应用客户程序的 OPC 自动化接口（OPC OLE automation interfaces）。COM 接口效率高，通过该接口，客户能够发挥 OPC 服务器的最佳性能，采用 C++ 语言的客户一般采用自定义接口方案；自动化接口使解释性语言和宏访问 OPC 服务器成为可能，采用 VB 语言的客户一般采用自动化接口。自动化接口简化客户应用程序的实现，但运行时需要进行类型检查，减慢了程序的运行速度。OPC 自定义接口是服务器必须提供的，而自动化接口则不一定提供[3-4]。

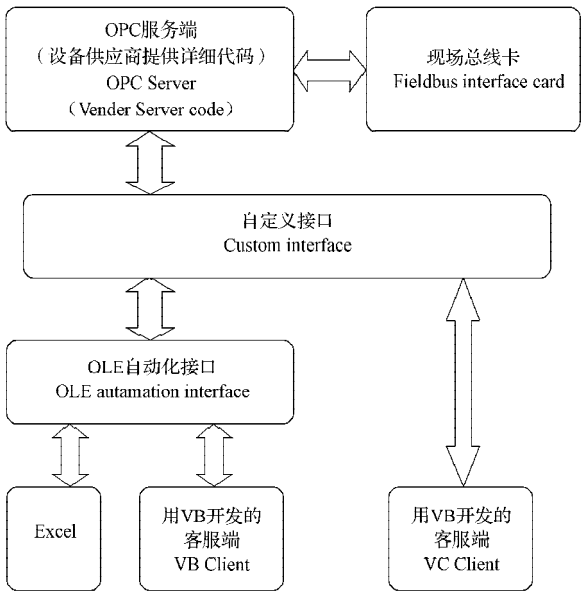

图 1　OPC 开发接口示意图

Fig. 1　The schematic diagram of the OPC interface

近几年，随着 Microsoft 的.Net 框架平台战略的成功，越来越多的企业在应用程序开发时都转向.Net 框架平台。基于 C # 语言的特性，且有较多的 OPCServer 供应商是不提供自动化接口的，所以在实际应用中最好采用自定义接口的方式开发。针对.NET 框架平台，OPC 基金会提供了 OPCRCW 动态链

接库和OPCNetAPI，用于在托管代码平台下的开发（属于自定义接口开发方式）。在.NET平台下，最方便的是OPC基金会提供的API类库。在其它平台下（如VB，EXCEL等），因为开发平台的特性，只能选用自动化接口。

OPC应用程序的一般开发步骤是：建立连接、建立OPC数据项目、数据的读写，下面通过不同工业场景应用实例介绍OPC应用程序的开发。

3 工业场景应用实例

3.1 能源管理系统数据交换

在能源管理及配电管理系统中，有的现场设备只提供串口通信，PLC无法连接上，而上位监控系统一般是用工业组态软件开发，主要支持主流的PLC通信驱动程序。在这种情况下，可以开发一个外挂的应用程序，通过串口通讯与现场设备信息交互，以监控组态软件作为OPC服务端，应用程序通过OPC与监控系统信息交互。在这里，OPC客服端应用程序充当了数据采集适配器（网关）的角色，程序结构如图2所示。

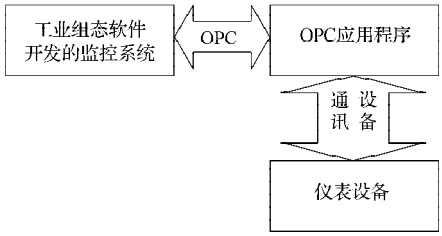

图2 OPC网关示意图

Fig. 2 The schematic diagram of OPC gateway

海南卷烟厂的能源管理系统中，有的仪表设备无法通过PLC直接通讯，笔者采用了上述外挂应用程序的信息交互方式。因所连接的仪表数量较少，且上位监控软件WinCC提供了自动化接口，故采用自动化开发方式。选用的开发平台为Microsoft Visual Studio 2008，需采用OPCSiemensDAAutomation.dll[4]。其开发步骤如下：

步骤1 建立连接

```
_OPCServer.Connect("OPCServer.WinCC", (object)
_computeName)[5];
```

步骤2 设置变量组属性

```
_OPCServer.OPCGroups.DefaultGroupIsActive=true;
_OPCServer.OPCGroups.DefaultGroupDeadband=0;
_GroupZTQ=
_OPCServer.OPCGroups.Add((object)"ZTQ");
_GroupZTQ.UpdateRate=2000;
_GroupZTQ.IsSubscribed=true;
```

步骤3 添加数据项目

```
_ItemsZTQ[1]=
_GroupZTQ.OPCItems.AddItem("1SJ_ZTQ",1);
_ItemsZTQ[2]=
_GroupZTQ.OPCItems.AddItem("2SJ_ZTQ",2);
_ItemsZTQ[3]=
_GroupZTQ.OPCItems.AddItem("SB_ZTQ",3);
_ItemsZTQ[4]=
_GroupZTQ.OPCItems.AddItem("TV_ZTQ",4);
_ItemsZTQ[5]=
_GroupZTQ.OPCItems.AddItem("1SL_ZTQ",5);
_ItemsZTQ[6]=
_GroupZTQ.OPCItems.AddItem("2SL_ZTQ",6);
_ItemsZTQ[7]=
_GroupZTQ.OPCItems.AddItem("3SL_ZTQ",7);
_ItemsZTQ[8]=
_GroupZTQ.OPCItems.AddItem("4SL_ZTQ",8);
_GroupZLP=
_OPCServer.OPCGroups.Add((object)"ZLP");
_GroupZLP.UpdateRate=2000;
_GroupZLP.IsSubscribed=true;
```

步骤4 注册事件

```
GroupOut.DataChange+=new
DIOPCGroupEvent_DataChangeEventHandler
(GroupOut_DataChange);
```

所注册的事件处理程序就是作数据处理的地方。

步骤5 数据处理

数据处理段就是和具体业务相关的处理程序，比如在本项目中，就是当在监控系统（WinCC）里点击命令按钮后，OPC程序把服务器端（WinCC）的数据转换为指令写入远程仪表。

3.2 包装机上位监控系统

在烟草生产中，如打叶复烤、制丝生产等，由于生产工艺线长、设备多，相应的监控画面及数据量庞大，而工业组态软件的特点是开发效率高，但二次开发的扩展功能有限，所以中控系统一般采用工业组态软件开发。但在卷包生产中，因为是单机设备，且客户定制的需求比较多，工业组态软件用在这里就不适合了，此时可以通过OPC与底层控制器通讯，上位监控软件采用灵活的.NET框架平台开发。如图3是笔者开发的包装机监控画面，设备的运行信息通过OPC获取底层数据驱动。

在笔者开发的包装机（包括FOCKE，B1等）改造项目中，除早期的几台用工业组态软件作上位监

控系统外，其它全部通过 OPC 和底层通讯，上位监控软件用.NET 平台开发。因为整个监控系统都是自主开发，为保证系统的运行效率以及 OPC 功能的最大利用，本系统采用自定义接口开发方式，应用 OPCRCW 包装类库，下面介绍其开发步骤。

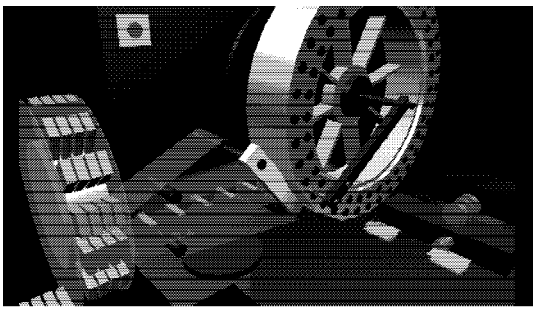

图3　包装机监控画面

Fig. 3　The monitor screen of packaging machine

步骤 1　建立连接

在开发环境中建立项目后，将动态链接库引入项目，就可编程实现对 OPC 服务器端的连接，具体代码如下：

```
//OPC 服务器对象类型名称
private const string  SERVER_NAME=
"OPCServer.XXX";
//OPC 服务器对象
private   IOPCServer pIOPCServer;
//OPC 对象类型
private Type svrComponenttyp;
// 获取 OPC 服务对象类型
svrComponenttyp =
Type.GetTypeFromProgID(SERVER_NAME);
// 创建 OPC 服务对象实例
pIOPCServer=(IOPCServer)Activator.CreateInstance
(svrComponenttyp) [3];
```

其中 pIOPCServer 是所要连接的服务器对象，SERVER_NAME定义了连接服务器对象的类型，这是为了方便记忆的一个名称，对应在注册表里注册的一个 CLSID，是 COM 对象的唯一标识字符串。接口 pIOPCServer 是在提供库中定义的，用户可直接调用。

在具体设计中，不要在代码里直接定义 OPC 服务器对象类型名称，而应该将它定义在项目的配置文件中，这样，当所连接的服务对象更改时，不用修改原代码，只要编辑系统配置文件即可。

OPC 数据项目（工业组态软件中的 TAG）采用分组的形式进行组织，为了对其数据进行访问，必须先添加组，然后在组中添加相应的数据项目。

步骤 2　OPC 数据项目的建立

1）添加组。具体实现函数封装如下：

```
public bool _AddGroup(IOPCServer _pIOPCServer,
string szGrpName,int iActive,int iClientGroup,int
iphServerGroup,Int32 dwRequestedUpdateRate)
```

其中：_pIOPCServer，组指向的服务器接口，由建立连接完成；

szGrpName，组的名称；

iActive，是否激活，1 表示激活该组，0 表示关闭该组；

iClientGroup，组的客户端句柄；

iphServerGroup，组的服务端句柄；

dwRequestedUpdateRate，组数据更新时间，单位 ms，如果该组数据实时性要求高，可将该值设为一较小值，反之则设为一较大值。

组添加成功，返回 true。

2）组中添加数据项目。具体实现函数封装如下：

```
public bool _AddItems(IOPCServer _pIOPCServer,
string szGrpName,int iCount,string [] zsITEMName, int []
ihClientHandle, Int16 [] iDataType)
```

其中：_pIOPCServer，组指向的服务器接口；

szGrpName，组的名称；

zsITEMName，添加数据的名称数组；

ihClientHandle，添加数据的客服端句柄数组；

iDataType，添加数据条目的类型数组；

组中数据项目添加成功，返回 true。

在具体的项目设计中，上面提到的这些参数（如组名称、是否激活、更新频率、变量 TAG 名称、变量类型，包括服务器名称等），可以在数据库中建立不同的配置表，所有参数都在系统启动时动态从数据库中加载，并设计一个单独的子程序，以友好的界面提供对这些数据的编辑。这样，当系统发生变化时，修改方便，可提高工作效率。

步骤 3　数据的读写

1）读取数据。在 OPC 组处于激活状态时，如数据发生变化，OPC 触发消息通知，用户通过订制该消息，建立消息接收函数，即可获得 OPC 中的数据当前值，消息的定义如下：

```
ValueChangeEventArgs(IOPCServer _pIOPCServer,
Int32 _hGroup,Int32 _dwCount,int[] _phClientItems, object
[] _pvValues,short[] _pwQualities,DateTime _DateTime)
```

其中：_pIOPCServer，组指向的服务器接口；

_hGroup，组的客服端句柄；

phClientItems，数据客服端句柄数组；

pvValues，数据值数组；

pwQualities，数据的质量代码；

DateTime，该消息的发生时间。

2）写入数据。通过函数 _WriteOPCData 写入数据，封装如下：

public bool _WriteOPCData(IOPCServer _pIOPCServer,Int32 _hGroup,int iCount,int [] iItemSvrHandle,Object [] objValue)

其中：_pIOPCServer，组指向的服务器接口；

_hGroup，组的服务端句柄；

iCount，写入项目的数量；

iItemSvrHandle，项目的服务端句柄数组；

objValue，数据值数组。

此类系统设计细节：作为一个监控系统，涉及的功能大概有系统数据、状态监控、系统参数设置、报警功能（当前和历史）、数据趋势查询（当前和历史）、数据统计（报表）功能。若仅需要在画面显示的变量，可以按画面分组，只有当前画面显示时才激活此组，而报警、趋势变量等，是要时刻都监视状态、保持数据的，这类数据组随时保持激活。参数设置等一般只要可以读写就可以，而且有的是保持在数据库中，这部分不用和 OPC 打交道，有的要求保持在 PLC 的数据块中，只要具备读写功能就可以。报警的设计可以分当前报警和历史报警，为避免频繁访问数据库，当前报警用内存表存储，当退出系统时做持久化处理。当前报警消除后，相应的记录移入历史记录表里。上述建议的核心思想是减少通讯负荷量，只有时刻需要数据变化的变量组一直处于激活状态，而其它变量只有在用户需要查看时才激活，这样可以大大减少实时通讯的数据量。

3.3 生产质量数据的实时监控分析

随着工艺技术的进步和对产品质量要求的提高，越来越多的企业要求对在线生产数据提供质量分析。但是组态软件的定位是快速开发监控系统，因其对扩展开发的局限性，对此类特殊的图表分析及大量数据处理难以胜任，一般可通过.NET 框架平台结合 OPC 技术开发实现。

目前有几款国外开发的统计过程控制（statistical process control，SPC）软件，但价格高昂，且相关分析功能不适合国内企业的定制要求。而此类应用软件完全可以通过 OPC 采集底层数据，结合相关数理统计知识自主开发。

此类系统的开发设计中，OPC 的主要职责就是数据采集与通讯。其它大部分的软件功能主要集中在数据的统计分析处理以及各种 SPC 图表分析上。因为上面已经讨论了 OPC 开发模式，这里就不再具体示例。SPC 软件功能不属本文讨论范围，也不在此详述。图 4 和图 5 是笔者开发的 SPC 质量分析管理软件的效果图。

4 结语

本文对 OPC 技术进行了简单介绍，探讨了 OPC 服务器与客户端的技术原理及客户端应用程序的设计方法。文中给出了能源管理系统数据交换、卷包生产控制、质量监控系统数据采集等开发设计实例。结果表明：OPC 技术大大提高了数据采集系统的开放性能，可以有效地避免开发过程的重复性，以及多种软件系统集成的不兼容性等问题。应用 OPC 技术可逐步实现软硬件之间的标准化，该技术在工业控制系统中有着广阔的应用前景。

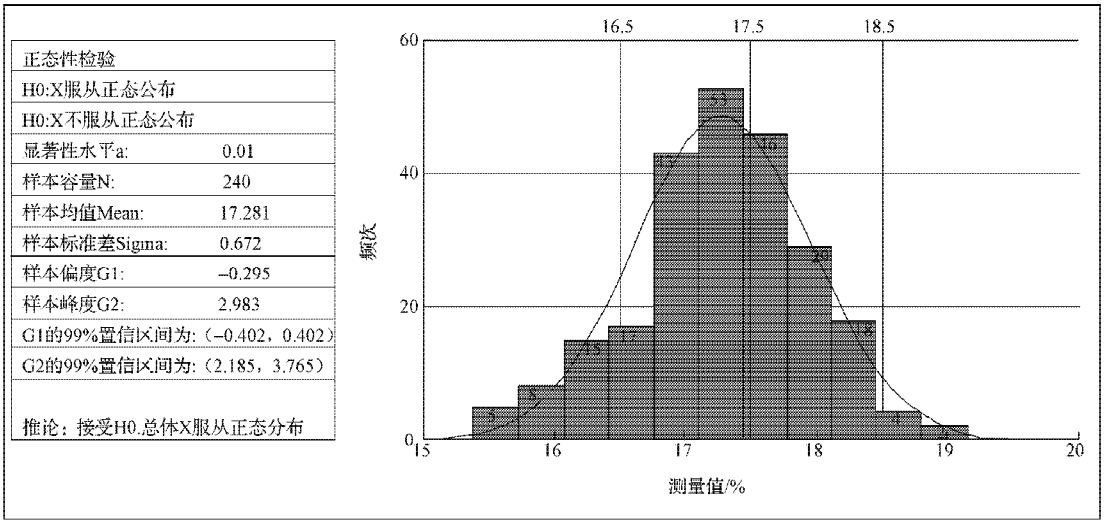

图4 直方图

Fig. 4 Histogram

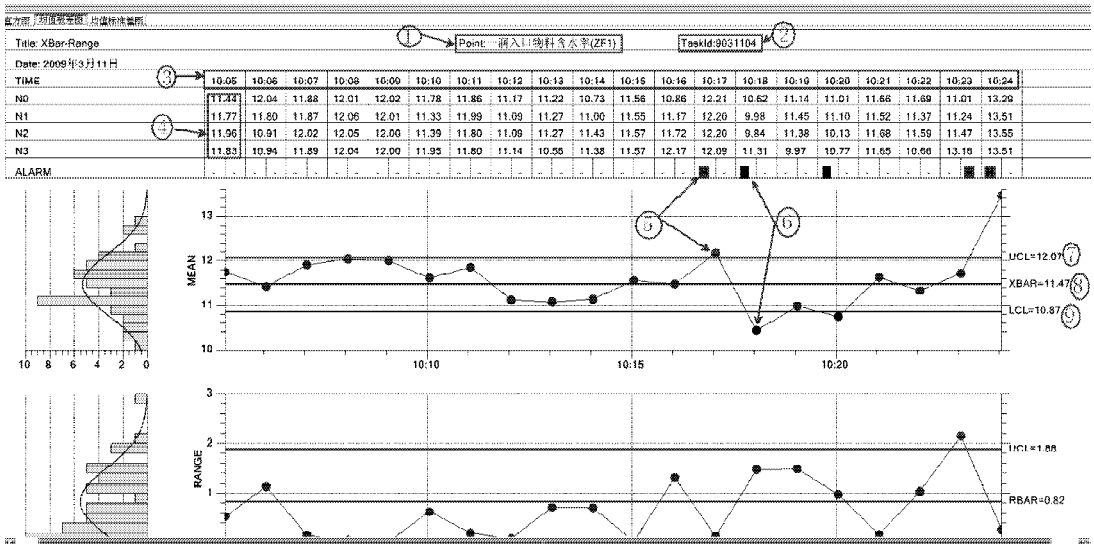

图5　均值 – 极差控制图

Fig. 5　The control chart of $\bar{X}$–R

参考文献：

[1] 吕正斌. 浅析 OPC 应用[EB/OL]. [2009-01-09]. http://www.gongkong.com/webpage/paper/200901/20090109130721000 01.htm.
Lü Zhengbin. A Brief Analysis of the OPC Applications [EB/OL]. [2009-01-09]. http://www.gongkong.com/webpage/paper/200901/20090109130721000 01.htm.

[2] Darek Kominek. OPC 通用指南：通向 OPC 的必经之路[M]. Alberta：MatrikonOPC，2009.
Darek Kominek. General Guidelines OPC：The Only Way to OPC[M]. Alberta：MatrikonOPC，2009.

[3] OPC 基金会. 数据访问自定义接口标准规范 2.0[S/OL]. [1998-10-14]. http://lhcb-online.web.cern.ch/lhcb-online/ecs/OPCEvaluation/htmlspef/index.html.
OPC Foundation. OPC Data Access Custom Interface Specification 2.0[S/OL]. [1998-10-14]. http://lhcb-online.web.cern.ch/lhcb-online/ecs/OPCEvaluation/htmlspef/ index.html.

[4] OPC基金会. 数据访问自动化接口标准规范2.02[S/OL]. [1999-02-04]. http://www-ad.fnal.gov/controls/opc/OPC_DA_Auto_2.02_Specification.pdf.
OPC Foundation. OPC Data Access Automation Interface Specification 2.02[S/OL]. [1999-02-04]. http://www-ad.fnal.gov/controls/opc/OPC_DA_Auto_2.02_Specification.pdf.

[5] 崔　坚. 西门子工业网络通信指南[M]. 北京：机械工业出版社，2006.
Cui Jian. Siemens Industrial Communication Guide[M]. Beijing：China Machine Press，2006.

（责任编辑：李玉珍）