面向串行通信的实时数据转储网关的设计与实现①

王昊轶[1] 赵国材[1] 季楠[2] (1.辽宁工程技术大学 电气与控制工程学院 辽宁 葫芦岛 125105;
2.南方航空公司机务工程部 沈阳飞机维修基地 辽宁 沈阳 110169)

摘　要： 针对在数据采集与监视系统（SCADA）的工程实践中，组态软件在处理通过串口传送来的数据帧时，由于通信协议不统一而存在的问题，提出采用VC++开发一个实时数据转储网关，完成通信协议解析和数据转储任务，解决通信协议不统一的问题。

关键词： 串行通信；实时数据；数据采集与监测系统；多线程

Design and Implementation of the Real-Time Data Dump Gateway for Serial Communication

WANG Hao-Yi[1], ZHAO Guo-Cai[1], JI Nan[2]
(1.School of Electrical and Control Engineering, Liaoning Technical University, Huludao 125105; 2. China Southern Airlines Aircraft Maintenance & Engineering Division, Shenyang Maintenance and Overhaul Base, Shenyang 110169, China)

Abstract: To address the disunity proplem of sheugang communication protocol when SCADA processes data frames from serial communication in the engineering of SCADA , a real-time data gateway developed with VC++ is proposed to complete the analysis of communication protocol and data dump.

Keywords: serial communication; real-time data; SCADA; multi-thread

1　引言

在数据采集与监视系统(SCADA)的工程实践中，通常使用来源于不同厂家的仪表设备采集现场数据，这些仪表设备再根据各自的通信协议，将采集到的实时数据封装成数据帧，通过串口传递给上位机进行处理，工业组态软件再将处理后的实时数据加以显示。在对这些采用不同协议的数据帧进行解析的时候，通常存在工业组态软件对数据帧无法解析的问题，这一问题在工程实践中显得尤为突出。

作者在目前承担的辽宁阜新金山煤矸石热电有限公司水源监测系统的项目实施中，根据系统的实时要求不高、监测变量变化缓慢的实际情况，设计开发了一个实时数据转储网关，采用松耦合的模式，将位于工业组态软件底层的，与数据帧的接收和处理有关的任务，从组态软件中分离出来，交由独立开发的实时数据转储网关来完成，这样就可以摆脱组态软件通信驱动的限制，使系统的开发更具独立性和自主性，很好的解决了以上问题。

2　项目分析

作者承担的项目需要对若干个位置分散、地理条件复杂的水源地进行监测。由于水源位置距厂区监控中心较远，因此数据通信采用无线传输的方式，即GPRS信号传输技术。现场仪表数据通过RS485传送给无线传输的B模块，B模块再通过GPRS无线网络传递给A模块，A模块通过RS232串口将数据帧传送给在监控中心的上位机中运行的组态软件加以显示，来达到对现场实时数据进行监测的目地，同时将

① 基金项目:辽宁工程技术大学研究生科研项目(Y200900404)
收稿时间:2009-10-21;收到修改稿时间:2009-12-12

实时数据存入数据库中供其它信息系统使用。其系统结构见图 1 所示。经过对以上的系统结构进行分析，可以采用三种方法解决上位机的数据接收和处理问题：

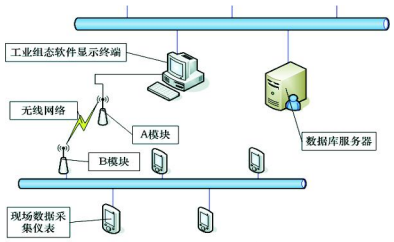

图 1　系统结构图

(1) 工业组态软件的通信驱动程序

组态软件厂家提供了部分仪表的通信驱动程序来实现数据帧的处理，但很不全面、很不完善，项目实施中受组态软件的约束太大，不够灵活。

(2) OPC 技术[1]

过程控制中的对象链接与嵌入(OPC)技术，此项技术以组件对象模型/分布式组件对象模型(COM/DCOM)技术为基础，在实现上较为复杂。

(3) 网关技术

采用高级程序设计语言开发一个完成协议解析和数据转储任务的网关软件，将实时数据转储到数据库中，组态软件通过访问数据库中已经解析完的数据来达到显示监测数据的目的。此种情况下的系统结构见图 2 所示。采用此种技术后，使系统处于松耦合的模式下，系统开发自主性更强，系统扩展更加灵活。

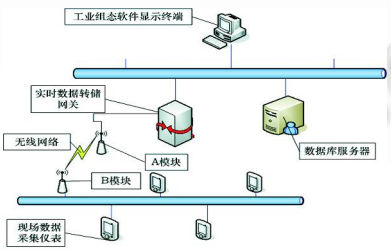

图 2　采用数据网关软件后的系统结构图

3　数据网关的架构设计

实时数据转储网关是一个具备通信协议格式自定义、数据帧实时解析和数据转储功能的软件。因此数据网关可划分为七个功能模块，分别为：用户接口模块、串行通信处理模块、通讯协议管理与解析模块、扫描线程模块、数据库处理模块、数据安全保护模块、数据存储模块。其系统架构见图 3 所示。

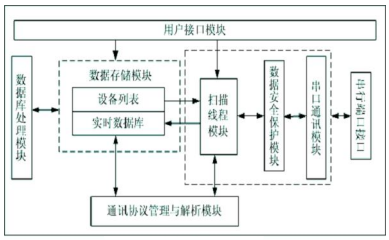

图 3　网关系统架构图

4　数据网关的模块设计

4.1　用户接口模块

此模块是同用户交互的图形接口，提供一个基于文档/视图技术的图形用户界面。

4.2　串行通信模块的设计[2-4]

串行通信处理器应该是一个基于多线程和重叠I/O(overlapped I/O)技术，集成对串口的访问操作，能够稳定的从串口收发数据，对外提供完善的使用接口，具有较好封装性的类。

在使用这个类时其工作流程是：首先设置串口通信参数，再开启串口监测工作线程。对于读数据，当这个线程监测到串口接收到数据、流控制事件或其它串口事件时，就以消息方式通知主程序，激发消息处理函数来进行处理。对于写数据，则是主程序发出写入消息，监测线程截获这个消息，执行写串口操作。其多线程机制如图 4 所示。

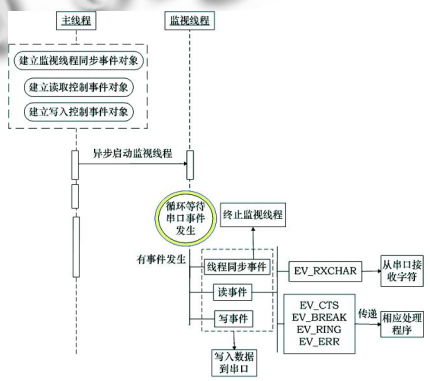

图 4　串口处理器多线程机制

所示的多线程机制可概括为以下四个功能：

(1) 串口设备的初始化：

在这里进行串口设备控制参数的初始化设置以及

用 CreateFile 这个 API 函数来打开串口，并且建立三个作为信号的事件对象，它们的初始态为未激活。其中读取事件由操作系统根据串口状态来控制，其它两个事件对象由主程序控制。

(2) 串口监视线程：

在完成串口设备的初始化后，通过调用 AfxBeginThread 函数来开启一个新的线程，用于监视串口发生的事件，根据不同的事件启动不同的处理程序。此处设置了一个处于无限循环的程序结构，通过调用 WaitCommEvent 函数来获取串口发生的事件，如有事件发生它将获取事件码并将读事件对象置为激活态。由于采用了重叠 I/O 操作，此函数调用后会立即返回，而读取事件对象不一定被激活，因此调用 WaitForMultipleObjects 函数进行等待，一旦读取事件对象、写入事件对象、监视线程事件对象中有一个被置为激发态或者等待超时，则停止等待进行相应的处理。

(3) 向串口写入数据:

在这里程序主动激活监视线程中的写事件对象，使监视线程在重叠 I/O 的模式下调用 WriteFile 这个 API 函数完成串口的写入操作。

(4) 从串口读入数据:

监视线程在重叠 I/O 模式下监测到 EV_RXCHAR 事件发生后采用临界区控制技术调用 ReadFile 这个 API 函数读取到达端口的字符数据。

4.3 通讯协议管理与解析模块

该模块分为协议定义和协议解析两部分：

协议定义器使用户可以根据项目的实际需要来建立通信协议格式数据库，供协议解析器分析数据帧时使用，它使数据网关具有较强的可扩展性。其界面结构如图 5 所示。

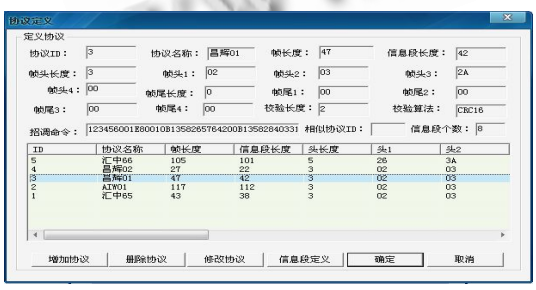

图 5　协议定义界面

协议解析器是能够根据协议定义库中的通信协议格式完成数据帧的解析的类。

4.4 扫描线程模块

利用 Windows 程序设计的消息响应，事件驱动机制，对从设备列表中读取出的招调命令按照招调策略(主要是对通信中断、通信阻塞、仪表无响应、数据校验错误的处理策略，在实现这些策略时要注意灵活运用 Windows 定时器)进行循环遍历招调，完成现场仪表监测数据的招调功能，并将数据写入实时数据库中。

4.5 数据库处理模块

数据库访问处理器从实时数据库中得到现场数据并将其保存到数据库中。

4.6 数据存储模块

该模块包括设备列表和实时数据库两个部分。设备列表存储用户对于设备的配置信息；实时数据库用于存储设备各通道实时值，供数据库处理模块调用。实时数据库驻留于内存中，有效保证网关的实时性。实时数据库的逻辑结构如下：

```
//实时数据库
typedef struct DeviceRTValue
{ CString deviceType; //设备类型
CString commandID; //招调命令在协议库中的 ID
CString deviceCommand//招调命令
float AIRTValue[80]; //模拟量实时值
long DIRTValue[240]; //开关量实时值
WORD ComIsErr; //设备通信状态
long LastScanTime; //设备上次扫描时间
long ScanPeriod; //设备扫描周期
BOOL IsSim; //是否仿真
} DEVICERTVALUE;
```

4.7 数据帧安全保护模块

该模块将收发的数据帧，按照设计好的算法进行解密和加密，以保证数据的安全性。

5 基于消息响应的收发机制实现

在以上设计的七个功能模块的基础上，充分利用 Windows 程序设计的消息响应，事件驱动机制实现了串口数据的收发，其工作流程和代码片段如下：

(1) 启动网关服务：

```
//设置内部定时器
::SetTimer(GetSafeHwnd(),RUN_TIMER,1000,
TimeTransData ); ……
```

////利用串行通信处理器初始化串口设备，启动串口监视线程

if(m_SericalPortU.InitPort(m_pRealTTranUDlg,m_nportnr,m_nbaud,'N',m_ndatabits,m_nstopbits,EV_RXCHAR,512))

{m_bSerialPortOpended=TRUE;

m_SericalPortU.StartMonitoring();} .....

// 设定重传定时器，MAX_TIME 毫秒后无数据到达，启动重传策略

::SetTimer(m_pRealTTranUDlg->GetSafeHwnd(), RC_TIMER, MAX_TIME, ReComunication);

///以招调命令 ID 为参数发出写命令消息，_commandID 为

//命令 ID

::SendMessage(m_pRealTTranUDlg->GetSafeHwnd(),WM_USER_WRITECOMMAND,(WPARAM)_commandID, (LPARAM) 0);

(2) 等待串口字节数据到达事件发生，接收数据：

////当串口有字节到达时监视线程会发出///WM_COMM_RXCH 消息,收到的字节作为消息参数传出

::SendMessage((port->m_pOwner)->m_hWnd,WM_COMM_RXCHAR,(WPARAM)RXBuff,(LPARAM)port->m_nPortNr);

////以下代码为 WM_COMM_RXCH 的响应函数 OnComm 中///的代码

////将接收到的字节存入缓冲区

pApp->m_ProtocolParse.m_strBuff=pApp->m_ProtocolParse.m_strBuff+(char)ch; .....

//////根据帧头判断协议格式

if(!pApp->m_ProtocolParse.SerchProtocol(zh_head)) {//无此

//协议

pApp->m_pRealTTranUDlg->OutPutData(" 数据帧采用了系统无法识别的协议!");

ClearComunication(&iLeng,&iWX_Count);return 0;}.

//以当前协议 ID（_protocolID）为参数发出

// “WM_USER_TRANSOVER”消息，通知网关处理缓冲区//里的数据帧。

::SendMessage(pApp->m_pRealTTranUDlg->GetSafeHwnd(),WM_USER_TRANSOVER,(WPARAM)_protocolID,(LPARAM) 0);

(3) 响应 WM_USER_TRANSOVER 消息完成数据帧的解析处理并将处理后的数据存入数据库，启动下一命令的招调。

/////关掉重传定时器

KillTimer(RC_TIMER); ……..////将缓冲区里的数据帧交给帧处理函数处理。

if(!pApp->m_ProtocolParse.ParseProtocolPacket(pApp->m_ProtocolParse.m_strBuff)){OutPutOperateLog("有效载荷数据校验错误！");

pApp->m_ProtocolParse.m_infoBuffParsed.clear();

///启动重传

pApp->m_ProtocolParse.ReCommand(); return FALSE;}

./////将解析后的数据存入数据库中

WriteToDataBase(); ……/////发出下一条招调命令

::SetTimer(GetSafeHwnd(),RC_TIMER,MAX_TIME,ReComunication); ::SendMessage(pApp->m_pRealTTranUDlg->GetSafeHwnd(),WM_USER_WRITECOMMAND,(WPARAM)_commandID,(LPARAM) 0);

6 实验与结论

为了验证网关的性能，制定如下实验方案并在实验室搭建了实验平台如图 6 所示：

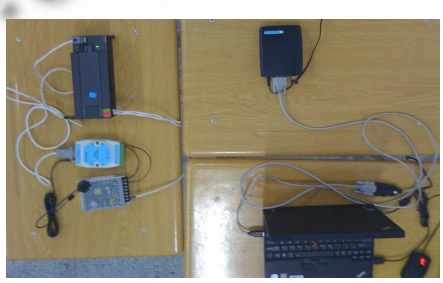

图 6 系统测试平台

采用西门子 PLC S7-200 与无线 DTU 的 B 模块相连，用 PLC 来仿真现场仪表作为采集系统的下位机；将无线 DTU 的 A 模块与运行网关软件和组态软件的计算机相连作为采集系统的上位机。

经运行测试，数据网关软件运行稳定，组态软件能

够实时显示现场数据见图 7、8 所示。

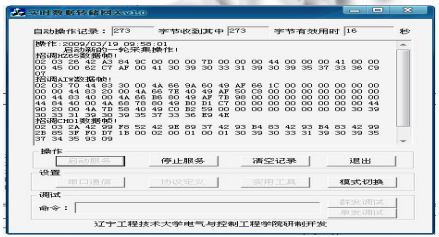

图 7　数据网关运行主界面

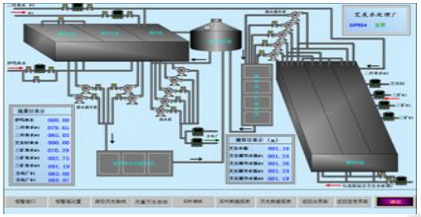

图 8　组态软件运行画面

7　结语

这一实时数据转储网关软件本着通用性强、灵活度高、易于扩展的思想进行规划和设计，达到了工程实际要求。同时由于实时数据被转储到数据库中，其它信息系统也就可以很方便的访问实时数据，也就解决了信息网络中的数据接口问题。此软件在作者承担的工程项目中所采用收到了良好效果。

参考文献

1　王锦标编著.计算机控制系统.北京:清华大学出版社, 2004.199－200.

2　Denver　A. Serial Communications in Win32. Microsoft Windows Developer Support, December 11, 1995.

3　侯杰译.Win32　多线程程序设计.武汉:华中科技大学出版社, 2002.278－300.

4　龚建伟,熊光明. Visual C++/Turbo C 串口通信编程实践(第 2 版).北京:电子工业出版社, 2007.87－128