第 32 卷 第 19 期 计 算 机 工 程 2006 年 10 月
Vol.32 № 19 Computer Engineering October 2006
·工程应用技术与实现· 文章编号：1000—3428(2006)19—0245—03 文献标识码：A 中图分类号：TP316.4

.NET Remoting 及其在 SCADA 主站系统中的应用

李玉凯，朱有产，秦金磊
(华北电力大学信息与网络管理中心，保定 071003)

摘 要：阐述了新一代的分布式应用技术.NET Remoting 的体系结构及其在分布式应用中的特点。以在 SCADA 主站系统中的应用为例，利用.NET Remoting 技术代替传统的 DCOM 组件以实现软数据总线的功能，解决了 Windows 操作系统环境下一直存在的“DLL 地狱”问题，提高了 SCADA 主站系统的灵活性和可扩展性。最后给出了用 C#.NET 实现的具体方法和步骤。

关键词：.NET Remoting；DCOM；分布式系统；SCADA；C#.NET

.NET Remoting and Its Application in SCADA Master Station System

LI Yukai, ZHU Youchan, QIN Jinlei
(Center of Information and Network Management, North China Electric Power University, Baoding 071003)

【Abstract】This paper introduces .NET Remoting system structure, which is a new generation of distributed application technology, and its features in distributed application. Taken the application in SCADA master system for an example, it purposes a new method replacing traditional DCOM component with .NET Remoting technology so as to realize the function of soft data bus. As a result, this method solves the “DLL Hell” problem, which has been existing under Windows environment and enhances the flexibility and scalability of SCADA master station system. It presents its implementation method and procedure with C# .NET.

【Key words】.NET Remoting; DCOM; Distributed system; SCADA; C#.NET

计算机和网络技术的不断进步推动了分布式技术的发展。通常远程访问方法主要有 CORBA、DCOM 和基于 Java 的 RMI 技术，它们都有其独特的特点，但也存在不同的技术缺陷。

分布式组件对象模型(DCOM)代码的重用性提供了一种模块化和面向对象的技术标准。它定义了定位和辩识其它组件功能的标准方式，并与编程语言无关。但是，DCOM 生来就具有很多难以解决的问题。如 DCOM 组件编程不易且与编程语言有关，另外 DCOM 的部署比较困难，新旧版本必须保持兼容，否则会产生所谓的“动态链接库(DLL)地狱”问题。

.NET 技术是微软公司推出的下一代平台技术，其中以.NET Remoting 取代以往的 DCOM 技术。.NET Remoting 是一个内涵丰富、可扩展的框架，为创建和扩展通过远程对象进行交互的分布式应用程序提供了灵活的模型。同时.NET 组件提供了方便的编写平台和丰富的类库资源，并且使用了程序集概念，解决了 Windows 环境下一直存在的“动态链接库地狱”问题。

在此，以.NET Remoting 技术在数据采集与监视控制系统 SCADA(Supervisory Control and Data Acquisition)主站系统中应用为例，介绍其实现过程。

1 .NET Remoting 分布式应用体系结构及原理

.NET Remoting 是一种分布式对象技术，是在.NET 框架中执行进程间通信的方式，它允许运行在另一机器上的应用程序、进程或者对象访问某机器上运行的对象，让远程对象看上去像本地的一样。它用于网络上不同计算机中基于 CLR 的不同应用程序之间的通信，也可用于同一台计算机中基于 CLR 的不同应用之间的通信。

.NET Remoting 的体系结构如图 1 所示：客户端应用程序域和服务端应用程序域的分界线构成.NET Remoting 的边界。服务端应用程序域由传输信道、序列化格式器和服务端对象组成，客户端应用程序域由客户端对象、代理、传输信道、序列化格式器组成。应用程序域是进程中新的安全分界线，可将它看作是一个逻辑进程。客户代理分为透明代理和真实代理。其中，透明代理是远程对象的精确副本，真实代理接受透明代理创建的消息并将其通过信道传递给远程对象。

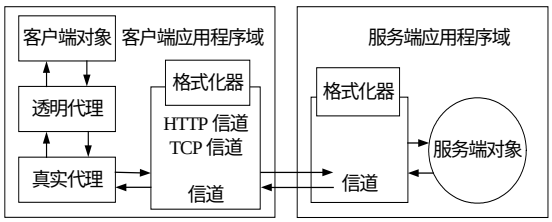

图 1 .NET Remoting 的体系结构

.NET Remoting 是基于进程间通信的机制，进程间通信需要一个向其进程外的调用方提供功能的服务端对象、一个在服务端对象上进行调用的客户端以及一个将调用从一端运送另一端的传输机制。客户调用服务端对象的方法要么创建对象的完整副本，并移动到客户端，即值调度；要么向客户端进程传递一个对服务端对象的引用，实现引用调度(MBR)。由于值调度占用很大的带宽，浪费客户端的内存和处理时间，

作者简介：李玉凯(1975－)，男，硕士生，主研方向：分布式系统，网格计算；朱有产，教授；秦金磊，硕士生

收稿日期：2006-01-05 **E-mail：**wslyk@sohu.com

且存在安全隐患，因此.NET Remoting 进程通信采用引用调度。为便于处理，.NET Remoting 向程序员提供了更为简单的处理过程，只需正确配置客户端，创建远程类型的新实例，而不像 DCOM 那样难以配置。

当客户端创建远程类型实例时，远程处理基础结构创建与远程类型完全相同的代理对象，并向客户端对象返回一个对该代理的引用。当客户调用此方法时，远程处理系统接收调用，检查类型信息，并通过客户信道将请求捆绑成消息路由到服务器信道。服务端侦听信道获取该请求并将其转发给服务器远程处理系统，服务器远程处理系统查找(必要时创建该对象)并调用服务器对象。反之，服务器远程处理系统将响应结果捆绑成消息发送到客户信道，客户端远程处理系统通过代理将结果返回给客户对象。

.NET Remoting 两个应用域空间连接的信道是跨远程处理边界在应用程序之间传输消息的对象。信道可以在终节点上侦听入站消息，并向另一个终节点发送消息，或双向执行。有 2 种信道：HttpChannel 和 TcpChannel，支持 HTTP、XML、SOAP 和 TCP 等多种协议。其中 HttpChannel 可承载 XML 文本型数据，TcpChannel 可承载二进制数据，因此 TcpChannel 较 HttpChannel 能得到更高的效率和速度。

.NET Remoting 不仅可以承载 IIS 的安全特性，也可自定义安全，甚至信道的传输协议。最后，为了使通信可行，必须注册信道和配置服务，客户服务必须激活应用才能得到.NET Remoting 提供的服务。

2 .NET Remoting 在 SCADA 主站系统中的应用

2.1 SCADA 主站系统

SCADA 系统是以计算机为基础的生产过程数据采集、传输与远程监视、控制调度自动化系统，广泛应用于工业自动化领域，一般采用分散式测控、集中式管理的方式。

SCADA 主站系统是 SCADA 系统的监控管理中心，为用户提供实时监控和操作界面，通常由通信前置机、服务器、多个监控工作站和将它们连在一起的局域网组成。各部分功能如下：

(1)服务器 安装在监控(调度)中心，运行一个较大规模的实时数据库系统，起实时数据汇集点的作用，是 SCADA 系统的中心。

(2)监控工作站 运行 SCADA 的人机界面，执行系统在线监控功能。工作站从服务器实时数据库取得实时数据显示，也通过服务器下发命令到远方。

(3)通信前置机 负责转换数据和通信协议，是服务器职能的延伸，起到 I/O 服务器的作用。通信前置机同时也在服务器和外部系统之间起隔离作用。

2.2 SCADA 主站系统数据传输流程

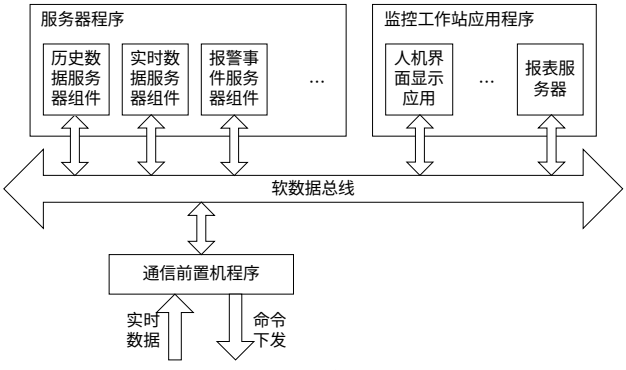

图 2 SCADA 主站系统数据传输流程

SCADA 主站系统中数据传送通过软数据总线实现，运行在通信前置机、各工作站和服务器上的软件模块都挂在此软数据总线上。在 Windows 系统下，以往常用的软总线技术是 DCOM，各软件模块通过 DCOM 接口进行进程间的通信和数据交互，见图 2。

2.3 .NET Remoting 技术在 SCADA 主站系统中的应用

在此以通信前置机程序向服务器程序传递数据为例。其具体方法是：在服务器程序中建立.NET Remoting 服务端远程对象，通信前置机和监控工作站程序中建立远程客户端。通信前置机程序将实时数据装入实时数据包，通过调用服务器程序的远程对象的 SendData()方法将实时数据包作为参数传递给服务器。同时，监控工作站程序通过调用服务端远程对象的 GetData()方法得到订阅的实时数据。具体步骤为：

(1)设计用于转载数据的数据包装类

数据包装类用来装载各种实时数据，是数据传递的载体。数据包装类含遥测数据队列、遥信数据队列等。由于此类实例要通过网络传输，因此必须可以序列化，方法是给此类加序列化属性。设计如下：

```
[Serializable]//可序列化属性
public class UpTransData//数据包装类
{…
  Private ArrayList remoteSignalList = new ArrayList();
//上行遥信数据队列
   private ArrayList remoteMeasureList = new ArrayList();
//上行遥测数据队列
    public UpTransData()//构造函数
    {…}
    //添加上行遥信数据和遥测数据的方法
    public void AddRemoteSignal(UpRemoteSignal data)
    public void AddRemoteMeasure(UpRemoteMeasure data)
    //获得所有遥信数据和遥测数据的方法
    public UpRemoteSignal[] GetRemoteSignal()
    public UpRemoteMeasure[] GetRemoteMeasure()
    …  }
```

最终将其编译成 UpTransData.dll 程序集，以便在后面的程序中引用。

(2)实现远程处理对象

.NET Remoting 服务端的实现首先要建立远程对象，为客户端提供远程访问。远程对象与其它类型用户对象惟一的区别是远程处理对象支持引用调度，因此远程对象必须从 MarshalByRefObject 对象派生。并将被远程访问的方法定义为公共的，并且不能是类的静态方法。远程对象定义如下：

```
public class AcceptData:MarshalByRefObject
{//静态缓冲区，传输数据都存入相同的缓冲区
  public static UpTransBuffer acceptBuffer;
  public AcceptData()//构造函数
  {//初始化数据包的内存缓冲区；
    acceptBuffer = new UpTransBuffer(size);
    …
  }
  //用于被前置机上的客户端远程调用的方法
  public void SendData(UpTransData data)
  {//将传输数据包存入内存中的缓冲区
    acceptBuffer.Save(data);
    …
```

```
}
//用于被监控工作站上的客户端远程调用的方法
Public UpTransData GetData ()
{…
   Return DaTa;
 }  }
```

(3).NET Remoting 服务的实现

.NET Remoting 服务是在服务器端运行的应用程序。它可以是控制台应用或 Windows 服务，也可以是一个 IIS 承载的应用，主要任务是发布能够被远程访问的对象。在服务器应用程序中创建.NET Remoting 服务，并以程序方式进行配置。在此需要引用程序集：System.Runtime.Remoting.dll 和前面创建的 UpTransData.dll。

1)创建 TCP 信道为服务器端的监听信道并指定其监听端口。一旦启动监听，TCP 信道将为每一个请求链接新建一个线程处理请求。创建 TCP 信道如下：

```
TcpChannel chan = new TcpChannel(8080);
```

2)注册通道。ChannelServices 处理提供注册和注销通道的服务，还可以在客户端把远程调用请求发到服务器，在服务器端调度传入的远程调用。代码如下：

```
ChannelServices.RegisterChannel(chan);
```

注册远程对象类型，可以指定远程对象中类的类型和模式。代码如下：

```
RemotingConfiguration.RegisterWellKnownServiceType(typeof(AcceptData),"AcceptData", WellKnownObjectMode.Singleton);
```

3)通过向远程处理系统注册，远程对象就加入了全局的远程处理表中。当调用到达服务器后，从消息中提取远程对象的标识信息，检查远程处理表以定位与该标识信息相匹配的对象的引用。

(4)客户端实现

客户端可以是控制台应用程序、Windows Form 应用程序，为了实现与远程对象的通信，必须添加对远程对象和.NET Remoting 名称空间的引用，并激活远程对象。其激活方式有客户端激活和服务器端激活。其中客户端激活客户可以控制对象的生存期。无论哪种方式都必须租约和续定生存期。客户服务层需引用前面的 AcceptData.dll 和 UpTransData.dll 程序集。客户端实现如下：

注册 TCP 信道，客户端注册 TCP 信道的时候不用指明信道的端口号且不在任何端口上侦听。但是，一定要保证客户端和服务器端有匹配的信道，否则会有异常出现。

```
ChannelServices.RegisterChannel(new TcpClientChannel());
```

获取远程对象，如果采用客户端激活则调用 Activator.CreateInstance()函数生成代理对象。采用服务器激活代理对象，代码如下：

```
obj = (AcceptData)Activator.GetObject(typeof(AcceptData),
"tcp://10.1.78.209:8080/AcceptData");
```

在通信前置机的客户端将上传的实时数据装入新建数据包装类的对象并通过调用远程对象的方法，将实时数据包作为参数传送给服务器模块：

```
Data = new UpTransData();
…
obj.SendData(UpTransData data);
…
```

在监控工作站的客户端，通过调用远程对象的方法得到实时数据包：

```
UpTransData data = obj.GetData();
```

最后，还需要在服务器上使用多线程来实现来自于多个客户机的并发请求，以一种安全线程的方式来开发远程对象。

3 结论

通过本文的介绍可以看出，.NET Remoting 技术具有很强的灵活性和扩展性，为处理局域网甚至互联网范围内的资源提供了一个绝佳的方法。在构建分布式 SCADA 主站系统的应用中，利用.NET Remoting 技术成功地实现了软数据总线功能，减少了设计难度，避免了使用 DCOM 技术所带来的软件部署和版本等问题，提高了系统的灵活性、易维护性和可扩展性，取得了较好的效果。

参考文献

1 Robinson S, Allen K. 杨　浩, 杨铁男译. C#高级编程[M]. 北京: 清华大学出版社, 2002.

2 张昆琪. Microsoft.NET Remoting 权威指南[M]. 北京: 机械工业出版社, 2003.

3 王常力, 罗　安. 分布式控制系统(DCS)设计与应用实例[M]. 北京: 电子工业出版社, 2004.

4 陈绪君. .NET 框架 Web Service 和.NET Remoting 分布应用解决方案及评价[J]. 计算机应用研究, 2003, 20(9): 110-112.

5 王正桓, 蔡　明. MS.NET Remoting 的分布式技术应用研究[J]. 计算机应用与软件, 2005, 22(3): 140-142.

（上接第 238 页）

3 结束语

本文所研究的 HT-7 数据采集控制系统在 2005 年春季 HT-7 物理实验中进行了首次应用。与以前的采集系统相比，新的系统在自动控制、稳定性和异常处理等方面都有了很大的改进和提高。从实际运行效果来看，也达到了设计的目标和要求。该系统的设计思想和控制流程也为 EAST(中科院等离子体所新一代的全超导 Tokamak 装置)在稳态放电实验条件下的实时数据采集控制进行了前期的准备和预演。

参考文献

1 Luo Jiarong. Towards Steady-state Operational Design for the Data and PF Control Systems of the HT-7U[J]. Nuclear Fusion, 2003, 43(9): 862.

2 Luo Jiarongl. The Distributed Control and Data System in HT-7 Tokamak[J]. IEEE Transactions on Nuclear Science, 2002, 49(2): 496-500.

3 朱　琳. HT-7 核聚变实验数据系统研究[D]. 北京: 中国科学院研究生院, 2004.

4 李　刚. 几种典型操作系统下基于 PCI 总线的数据采集研究[D]. 北京: 中国科学院研究生院, 2005.