一种CPS中C#与C++代码互访方法的研究

许 春,杨 彬,赵 辉,伍良富
(四川大学,四川 成都 610064)

摘 要:客户编程软件(CPS)是对讲机等无线通讯系统Radio配置系统的核心技术,实现C#与C++代码的互访是CPS设计的研究热点之一。提出了一种利用平台调用和回调函数的形式实现C#与C++的互访的方法。该方法既保持C#语言进行软件开发具有开发效率高、便于维护和管理的优势,又具有C++开发的系统执行效率高和健壮性高的特点,并且提供了对系统底层更灵活的访问的能力,该方法是一种Radio配置系统CPS的好方法。

关键词:无线通话系统;客户编程软件;公共语言运行时
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2007)03-0020-03

Research about Method to Implement Inter-accessing of C# Code and C++ Code for CPS

XU Chun, YANG Bin, ZHAO Hui, WU Liang-fu
(Sichuan University, Chengdu 610064, China)

Abstract: The Customer Programming Software (CPS) is the core technology of radio configuration subsystem of the wireless communication system such as wireless transceiver. One of the most essential problems is how to implement the inter-accessing of C# code and C++ code. A method of implementing the inter-accessing of C# code and C++ code by using Pinvoke and callback function is introduced in this article. This method could not only retain the advantage of C#, such as its high efficiency in developing application and its easy way of maintaining and management, but also remain the advantage of C++, such as its high efficiency in the running and its strong robustness. This method also provides the ability of accessing the low level of system in a more flexible way, so it is a good method for radio configuration system and CPS.

Key words: wireless communicating system; customer programming software; common language runtime

1 Radio及CPS简介

作为无线通话系统,Radio主要应用在公安、民航、运输、水利、铁路、制造、建筑、服务等行业,用于团体成员间的联络和指挥调度。在通信网络未覆盖的地方,对讲机成为紧急调度和集体协作的重要工具。但是,在正式使用Radio之前,必须根据实际情况对Radio的复杂参数进行配置[1]。

客户编程软件(Customer Programming Software, CPS)系统,是一种常用的Radio配置系统。CPS可以很方便地对Radio进行参数配置、保存、读取等,从而使得用户操作Radio变得更方便、简单和友好[2]。

在CPS中,存在如下的模块调用关系(见图1):

其中,GUI和中间层Controller均用C#编写,具有很高的开发效率。而下层通用通信组件(Common Communication Component, CCC)则由C++编写而成,属于经过验证的历史遗留代码,具有很高的可靠性、健壮性[3]。

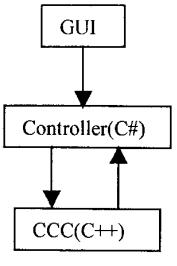

图1 CPS模块调用关系

CPS设计和开发中的一个重要特色是提供了从C#访问C++的代码,而从C#访问C++的代码备受人们关注。

公共语言运行时(Common Language Runtime)提供了两种机制访问非托管代码(C++代码)[4]:

收稿日期:2006-06-19
作者简介:许 春(1971-),男,河北卢龙人,讲师,博士研究生,研究方向为网络安全、编译技术;伍良富,教授,研究方向为网络安全、编译技术。

(1)平台调用,允许托管代码(C#)调用非托管代码导出的函数;

(2)COM Interop,允许托管代码与COM对象通过接口进行相互访问。

利用Pinvoke技术[5],设计了一种方法,以平台调用的形式来访问C++组件,并且通过回调函数实现C++组件访问C#。

2 Pinvoke技术

在CPS中,使用平台调用来允许托管代码调用以DLL形式实现的非托管函数,在必要时,对参数进行封送处理(Marshaling)。

平台调用倚赖元数据(Metadata)定位DLL中的导出函数并在运行时封送其参数,图2很好地说明了这一过程。

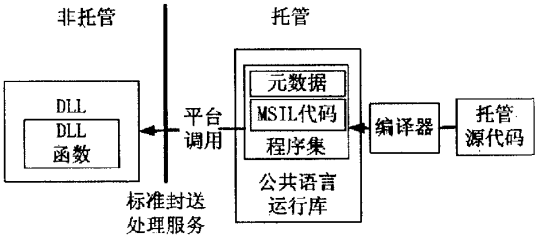

图2　托管代码到非托管代码的调用

从图中可以看到调用流是从托管代码到非托管代码。但是,在CPS中,一些非托管代码需要完成一些任务,比如LogApplicationError。在这种情况下,需要引入回调函数(委托)来实现,图3演示了更新后的调用流程。

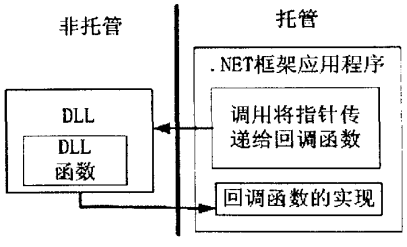

图3　非托管代码调用托管代码

3 从.Net代码到非托管代码的调用

3.1 标识Communication DLL中的函数

在C++编写的Communication DLL中,需要提供给C#使用的Open方法函数标识包含如下两个方面:

* 函数的名称,Open。

* 实现所在的DLL文件的名称,Communication.dll。

函数名称可以使用dumpbin－exports Communication.dll或link－dump－exports Communication.dll来获取。

在托管代码中,可以将非托管函数重命名为任何所需的名称,但是必须将该新名称映射到DLL中的初始入口点。

3.2 在托管代码中创建容纳非托管DLL函数的类

要封装平台功能,一种常用的方法是将DLL函数包装在托管类中。在一个内部类中,为每个要调用的DLL函数定义静态方法。定义中可以包括一些附加信息,如在传递方法参数时使用的字符集或调用约定。如果省略这些信息,将选择默认设置。

例如:

```
internal class WrapCommunication
{
[DllImport("Communication.dll", EntryPoint = "Open", CharSet
= CharSet.Auto]
public static extern uint Open(uint uiIPAddress,
  uint uiProtocolType,
  uint uiProtocolPortNumber,
  out uint uiHandle);
}
```

这里DllImport属性指定了包含该方法的DLL的名称,并通过EntryPoint指定了函数在DLL中的名字。通常的做法是在封装类中使用与导出的方法相同的名称命名C#方法,但也可以对C#方法使用不同的名称。

封装之后,就可以按照对其他任何静态函数调用方法的相同方式来对该函数调用方法。另外,为平台调用设计托管类时,应考虑类和DLL函数之间的关系,进行合理的分类,以便于记忆和使用。例如:在CPS项目中,可以定义WrapFirmwareUpgrade,WrapSerialTransport等多个封装类,以便于函数之间相互逻辑,便于查找。

3.3 在托管代码中创建原型

3.3.1 指定入口点

入口点用于标识函数在DLL中的真实名称。托管代码可以将入口点映射到一个不同的名称,比如前面的Open,可以根据需要重命名为OpenSerialTransport,以达到见名知义的目的。也就是说,可以在托管代码中将非托管方法重命名使用。

在前面的示例代码中,DllImportAttribute.EntryPoint就指定了DLL函数的真实名称。如果函数在方法定义中的名称与入口点在DLL的名称相同,则不必用EntryPoint字段来显式地标识函数。

在以下情况下,通常需要对非托管代码中的DLL函数重命名:

* 避免使用区分大小写的 API 函数名;

* 符合现行的命名标准;

* 提供采用不同数据类型的函数(通过声明同一 DLL 函数的多个版本);

* 简化对包含 ANSI 和 Unicode 版本的 API 的使用。

3.3.2 指定字符集

DllImportAttribute.CharSet 字段控制字符串封送处理并确定平台调用在 DLL 中查找函数名的方式。

CharSet 字段接受以下值:

* CharSet.Ansi(默认值)。

平台调用将字符串从托管格式 (Unicode) 封送为 ANSI 格式。

* CharSet.Unicode。

平台调用会将字符串从托管格式 (Unicode) 复制为 Unicode 格式。

* CharSet.Auto。

平台调用在运行时根据目标平台在 ANSI 和 Unicode 格式之间进行选择。

3.3.3 平台调用示例

例 1:C# 调用 C++。

```
internal class WrapCommunication
{
  [DllImport("Communication.dll",
  EntryPoint="Open",
  CharSet=CharSet.Auto]
public static extern uint Open(uint uiIPAddress,
  uint uiProtocolType,
  uint uiProtocolPortNumber,
  out uint uiHandle);
}
Public class Test
{
  private uint uiHandle;
  Public static void TestFunc()
  {
    //打开 192.168.0.2 的地址
    WrapCommunication.Open(3232235522,1,
    2006,
    out uiHandle);
  }
}
```

4 从非托管代码调用托管代码

在 C# 中,回调函数是通过委托(delegate)来实现的。委托可以把函数的应用与实现分离,允许实现作为独立模块存在。与函数指针不同的是,委托是类型安全的,这使得编译器和.Net 框架确保通过委托调用代码能够安全执行。

C# 中可以通过一个两步过程来申明委托:首先申明委托类型;然后创建委托类型的实例。

下面给出 CPS 中的一个实例:

例 2:C++ 调用 C#。

```
[return: MarshalAs(UnmanagedType.U4)]
public delegate uint DLogAppErr(
[MarshalAs(UnmanagedType.U4)]uint Error,
[MarshalAs(UnmanagedType.BStr)]string Component,
[MarshalAs(UnmanagedType.BStr)]string Desc);
public class CDatabaseController
{
  [DllImport("communicationadapter.dll", EntryPoint = "OpenRC")]
  public static extern uint Open(
    IntPtr pSerialTransport,
    uint uiHandleToSerial,
    [MarshalAs(UnmanagedType.FunctionPtr)] DLogAppErr LogError,
    uint TargetRadio,
    out uint uiHandle);
  public void Func()
  {
    DLogAppErr logAppErr = new DLogAppErr();
    ......
    objRadioCodeplugController.Open(parm1,param2,
      m_objLogAppErr,
      param3);
  ......
  }
  [return: MarshalAs(UnmanagedType.U4)]
  public uint LogApplicationError(
    uint uiError,
    [MarshalAs(UnmanagedType.LPTStr)]string strComponent,
    [MarshalAs(UnmanagedType.LPTStr)]string strDescription)
    {
      //Implementation
    }
}
```

5 Interop 封送处理

大多数数据类型在托管和非托管内存中都具有公共的表示形式,Interop 封送拆收器可以自动处理这些类型,其他类型可能是不明确的,或根本不在托管内存中表示。

对于不明确的类型,需要进行显示的封送处理

(下转第 47 页)

表 1　FOE 法实验结果

坐标	交点一	交点二	交点三	交点四	平均值
C_x	387.76	386.52	392.07	384.44	387.698
C_y	294.85	299.31	294.70	296.16	296.255

面模板固定在一个刻度精确的轨道上,标定模板和图像如图 5 所示,使摄像机光轴尽可能垂直平面模板,在摄像机的有效焦距范围内通过调节摄像机镜头光圈,使图像清晰,焦距为 f 时采集图像 1,调节光圈,改变焦距到 f',得到图像 2,通过图像处理,得到图像上相应特征点(各圆的圆心)的像素坐标,采集到的数据经过计算和最小二乘法拟合,解得图像中心坐标(c_x,c_y)。利用最小二乘法,得出图像中心坐标为(393.391,279.688)。计算结果综合比较如表 2 所示。

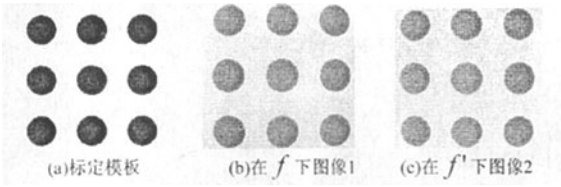

图 5　验证实验模板及图像

表 2　各种方法综合比较

坐标	径向准直法	求 FOE 法	变焦距法	线性模型标定分解	图像帧存中心
C_x	398.356	387.698	393.391	375.619	383.5
C_y	302.875	296.255	279.688	299.397	287.5

3　总　结

文中讨论的各种方法中,直接光学方法原理简单,而且重复性和精度最好,但由于其实验设备和条件相对苛刻,在精度要求很高的视觉测量场合中才用到。径向准直约束法的精度也较高,但是方法原理比较复杂,限制相对较多,计算也比较复杂,该解法一般仅使用大畸变镜头的情况。由于它是一种基于径向准直约束和等比不变性的方法,若畸变很小或没有畸变时,解算过程可能不稳定。变焦距法和求延伸焦点(FOE)法原理相对简单,操作容易,实验条件要求不高,精度相对较低,对于一般测量场合值得推广。

参考文献:

[1] Lenz R K, Tsai R Y. Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3 - D Machine Vision Metrology[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 1988, 10(5): 713 - 720.

[2] 苏小华,赵继广,张慧星. CCD 摄像机成像畸变的研究[J]. 物理实验,2003,23(9):64 - 68.

[3] 周士侃,邢　渊. 用于反求工程的 CCD 摄像机图像中心、比例因子的标定[J]. 计量与测试技术,2004(6):29 - 36.

[4] 李德仁,王新华. CCD 阵列相机的几何标定[J]. 武汉测绘科技大学学报,1997,22(4):70 - 76.

[5] 郑南宁. 计算机视觉与模式识别[M]. 北京:国防工业出版社,1998:26 - 27

(上接第 22 页)

——将 DllImportAttribute 属性应用于托管代码中的静态函数或方法,并用托管数据类型替换非托管数据类型。常用类型转换见表 1。

表 1　常用类型转换表

非托管类型	托管类名	说明
BStr	System. String	8 位
Bool	System. Boolean	32 位
……	……	……

示例代码:

```
[DllImport("communication")]
public static extern uintIoControlBOOL(
  uint uiHandle,
  [MarshalAs(UnmanagedType.BStr)]string strCmd,
[MarshalAs(UnmanagedType.Bool)]ref bool VARIANT_BOOL);
```

在这里,由于 uint 类型在托管和非托管代码中具有同样的表示形式,无须进行显式封送处理。而对于 System. String 和 System. Boolean,则需要进行 Interop 封送处理,并遵循转换规则进行操作。

6　结　论

随着 Radio 的应用领域的扩大,对 Radio 的复杂参数进行配置,特别是批量配置,成为业界研究热点。

在 Radio 配置系统 CPS 中充分利用 Pinvoke 技术(Interop),在 C# 工程中重用成熟的 C++ 代码,不仅能减少重复劳动,提高软件生产率,缩短开发周期,而且能有效提高软件系统的质量。

参考文献:

[1] 周　毅,邵　晖,金　玮. 基于虚拟仪器的车载收音机在线测试系统[J]. 计算机测量与控制,2004(12):1159 - 1161.

[2] 刘　鑫. 中国 CompactPCI/PXI 技术发展与应用[J]. 测控技术,2004,23(6):4 - 7.

[3] 黄瑾娉, Windows 监控程序的核心编程原理及设计方法[J]. 微机发展,2004,14(11):87 - 89.

[4] Stevens A I, Walnum C. 标准 C++ 宝典[M]. 北京:电子工业出版社, 2001.

[5] 胡智文,陈国龙. 在 Visual Studio. NET 中有效利用 Windows API 资源[J]. 计算机工程与应用,2004(12):108 - 111.