

基于 CORBA 的数控机床 DNC 模块开发

张爱红

(无锡职业技术学院 机电系, 江苏 无锡 214121)

摘要: 为了保证柔性制造系统 (FMS) 高效、可靠地运行, 离不开计算机对数控机床的实时控制 (DNC)。无锡职业技术学院数控机床网络 DNC 模块在 CORBA 统一通信平台的基础上实现了与仿真调度等模块的通信及分布式控制, 该模块主要包含 3 个子模块, 即: “通信接口模块”、“DNC 功能模块”、“PLC 控制模块”, 文中阐述了这 3 个功能模块开发过程, 并给出了具体的开发实例。

关键词: CORBA; 通信; DNC; 控制

中图分类号: TH165 **文献标识码:** A **文章编号:** 1672-0318 (2006) 04-0010-04

现代网络的多样性使得网络编程变得较为困难, 而通过 IONA 公司开发的 Orbix 软件编程环境, 基于 CORBA 协议来开发面向对象的分布式程序, 将大大缩短程序开发的周期并降低开发的难度。无锡职业技术学院 FMS II 期中程序的开发也基于 CORBA 通信平台, 使分布在不同计算机上的模块能够相互通信, 以达到分布式控制的目的。本文针对数控机床 DNC 模块, 及其与仿真模块、PLC 控制模块的通信编程思路, 给出程序开发的详细过程。

1 通信接口模块的开发

Common Object Request Broker Architecture 简称为 CORBA, 是一种标准或协议。它为开发面向对象的、分布式应用程序提供了一个框架, 而对象请求代理 (Object Request Broker, ORB) 则是 CORBA 的核心^[1]。

1.1 ORB 简介

ORB 实质上是一个软件组件, 在功能上充当客户程序与服务器程序间通信的介质。通过 ORB 组件可以隐藏网络编程的复杂性。ORB 对象实现位于服务器上, 其成员函数可被位于网络上的客户程序调用。当客户调用 CORBA 对象成员函数

时, ORB 截取该调用, 并借助于网络将之重定向到目标对象, 由 ORB 接受功能调用的结果并返回到客户端, 如图 1 所示^[1]。

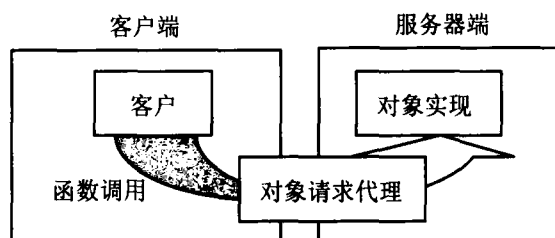


图 1 对象请求代理 (ORB)

CORBA 对象属于标准的软件对象, 可由 Visual C++ 语言来实现。每个对象都有明确定义的接口, 通过接口定义语言 (IDL) 来完成。为了调用 CORBA 对象的成员功能, 客户只需要知道对象的 IDL 定义, 无需了解如何实现对象的细节或者对象在网络上的位置, 因此可以实现对象接口与对象实现的分离。

1.2 接口的定义

程序开发首先要定义系统中的对象接口, 然后使用 IDL 编译器对生成的 IDL 代码进行编译。编译的结果将由 IDL 代码生成一组 C++ 代码, 包括 2 个部分: ① 客户存根代码 (CSC: Client Stub Code), ② 对象框架代码 (OSC: Object Skeleton Code)。

在本应用程序中, 用记事本来创建文本文件,

收稿日期: 2006-08-18

项目来源: 江苏省高校高新技术产业发展指导性项目 (FMS 二期)

作者简介: 张爱红 (1971-), 男, 江苏盐城人, 副教授, 硕士, 研究方向为机器人、数控等机电一体化技术。

保存时将扩展名更改为 idl。接口变量与函数的定义举例如下:

```
interface fanucOi
{ .....
    attribute short machStatus;
    readonly attribute string ncProcID;
    void machStartFile(in string ncProcID);
    .....};
```

变量 machStatus 取值的不同代表机床状态的不同, 0: 复原, 1: 运行, 2: 停止; 变量 ncProcID 定义 DNC 程序名, readonly 表示 ncProcID 为只读变量, 而函数 machStartFile(in string ncProcID) 的调用可以启动名为 ncProcID 的数控程序 DNC 加工。

接口定义后在命令提示符下键入“idl -B fanucOi.idl”, 将编译生成 3 个文件: fanucOi.hh, fanucOiC.cpp, fanucOiS.cpp, 这些源文件包含了同 IDL 定义相对应的 C++ 定义, 分为客户存根代码(CSC: fanucOi.hh, fanucOiC.cpp) 和对象框架代码(OSC: fanucOi.hh, fanucOiS.cpp)。其中: 数控机床 DNC 模块属于服务器端程序, 需将对象框架代码加入到工程中; 相应地, 客户存根代码需加入到客户端(仿真调度)模块中。

1.3 接口的实现

实现接口的方法有: BOAImpl 法与 TIE 法 2 种。以 BOAImpl 法为例, 服务器端程序开发分 3 步: ①首先定义一个 C++ 接口类: fanucOi_i。该类从 fanucOi.hh 中定义的 fanucOiBOAImpl 类继承而来, 接着给出类成员变量、成员函数的定义, 例如定义变量: CORBA::Short m_machStatus, 并给出实现代码, 将之与接口变量 machStatus 联系起来; ②创建对象的实例。在服务器程序中以全局变量的方式申明类实例: fanucOi_i fanucOi, 在循环线程函数中根据::fanucOi.m_machStatus 数值的不同来判断机床所处的状态, 执行机床运行监控程序或根据机床所处的状态来赋值; ③向客户程序公布 Orbix 对象。通过调用 Orbix 函数 CORBA::Orbix.impl_is_ready("fanucOi", 0L) 来完成初始化并处理来自客户的函数调用, 该函数包含两个参数: 服务器名及超时时间。如果有函数调用到达, 该函数返回, 如果出现意外, 程序捕获异常并提示用户服务器没有准备好。

客户端程序(仿真模块)的开发分为 2 步:

①使用绑定机制获得对象的引用。通过 fanucOi_var dVar 定义代理对象 dVar, 然后调用 fanucOi::_bind(":fanucOi", "fms"), 返回客户地址空间的代理对象 dVar, 其中“fanucOi”为服务器名, “fms”指明了服务器运行的主机名; ②绑定成功后, 即可访问 IDL 接口中定义的方法和操作, 例如调用 theApp.dVar->machStatus() 来判断机床的状态, 从而实现了仿真调度模块与机床 DNC 模块间的通信与互操作。

1.4 预处理操作

为了使程序顺利通过编译, 编译前需要完成:

①将 c:\ino\orbix_2.3c\inc 目录下的 corba.h 文件、corba 文件夹以及...orbix_2.3c\lib\itgi.lib 文件复制到 VC 开发软件的相关目录下; ②点击工程菜单->设置->C/C++ 标签, 添加预处理程序定义: WIN32, DEBUG, WINDOWS, _AFXDLL, _MBCS, ORBIX_DLL, IT_EX_MACROS; ③点击工程菜单->设置->连接标签, 添加 itgi.lib 对象/库模块; ④在命令行输入 :set IT_CONFIG_PATH C:\NONA\Orbix_2.3c\cfg 注册环境变量; ⑤在命令行输入:putit fanucOi(服务器名)_persistent 注册服务器。完成上述操作后, 接口模块即可正常进行编译、调试与运行。

2 DNC 模块的开发

为了实现 CNC 与上位机间的网络通信, 并考虑到通信的实时性, 需在 CNC 端安装 FAST ETHERNET 板、快速以太网控制软件并为 CNC 配置 IP 地址、子网掩码, 本模块中设置值分别为: 210.28.147.26、255.255.255.0, 使之与 FMS II 局域网处于同一工作组中。

FANUC 公司提供的 FOCAS2/ETHERNET 库函数具有非常丰富的功能, 可以通过 Visual C++ 编程调用, 实现: ①对机床门、液压卡盘等部件的开、关控制; ②数控加工程序的上传与下载; ③数控程序的 DNC 加工; ④数控机床复位操作等。

2.1 初始化操作

与建立 CORBA 通信程序的预处理操作相似, 在 DNC 模块的开发中需要进行一系列初始化, 以便能够调用封装的函数。包括: 将 FOCAS2/ETHERNET 中与通信相关的库文件 fwlib32.dll、fwlib32.lib, 与网络通信相关的头文件 fwlib32.h、库文件 fwlibe1.dll 复制到应用程序的文件夹中, 在 LINK 目标文件时需要为 LINK 程序指定 fwlib32.dll

的引入库 `fwlib32.lib`，通过点击应用程序“工程”菜单，出现“设置”子菜单后选中“Link”项，在“对象/库模块”中添加 `Fwlib32.lib`。为了能够调用 FOCAS2 中网络通信库函数，还需包含头文件 `fwlib32.h`。

2.2 通信句柄的取得与释放

网络通信前首先要取得通信句柄，调用语句：`cnc_allclibhndl3 ("210.28.147.26",8193,0, &h)`，其中：“210.28.147.26”为机床的 IP 地址，8193 为 DNC1/Ethernet (TCP) 端口号，第 3 个参数设定为 0，表示超时时间为无限，最后一个参数也就是调用该函数后取得的通信句柄。通过检查 `cnc_allclibhndl3` 的返回值是否等于 `EW_OK(0)` 来判断调用是否成功，在调试过程中曾一度出现返回值为 `EW_NODLL (-15)`，经分析后发现是由于没有调用与网络通信相关的动态库 `fwlibe1.dll` 所致。

一旦得到通信句柄后，其他函数都要将之作为参数传入。也就是它要伴随通信的全过程，直到调用函数 `cnc_freelibhndl(h)` 释放它为止。

2.3 开关量控制实例

数控机床中开关量的控制是通过 PLC 来实现的，在 Fanuc 系统中，有如下规定：机床→PLC 的信号为 X，数控系统→PLC 的信号为 F，这两类信号为 PLC 的输入信号，只能读，不能写；PLC→机床的信号为 Y，PLC→数控系统的信号为 G，这两类信号为 PLC 的输出信号，可以进行读、写操作。

以机床门的控制为例，门的状态信号分别为：`X4.1`（门开）、`X4.2`（门关）；门的控制信号分别为：`Y4.3`（关门）、`Y4.4`（开门）。在 FOCAS2/ETHERNET 库中，提供了进行机床 PMC 操作的函数，分别为：`pmc_rdpmcng`（读）、`pmc_wrpmcng`（写）。读写操作前，需定义 `IODBPMC` 类型的数据结构。读机床门状态的例程如下：

```
IODBPMC buf_door;//定义数据结构
short length;
buf_door.type_a=3;//定义地址类型：X 信号为 3，Y 信号为 2
buf_door.type_d=0;//定义数据类型：0 代表字节变量
buf_door.datano_s=4;//PLC 起始地址
```

```
buf_door.datano_e=4;//PLC 结束地址
length=9;//数据块的长度
pmc_rdpmcng(h,buf_door.type_a,buf_door.type_d,
buf_door.datano_s,buf_door.datano_e,length,&buf_door);//读 PMC 数据
short state=buf_door.u.cdata[0]&(0x06);
//保留第 1, 2 位数据，屏蔽其他数据位，得到门的状态数据
```

最后将 `state` 与 `0x02`（门开），`0x04`（门关）比较，从而判断出机床门的状态。液压卡盘、循环启动、机床复位控制程序开发过程与此相似，限于篇幅，不作介绍，详见 FANUC 公司开放数控系统 FOCAS1/2 库使用说明^[2]。

2.4 数控程序的 DNC 加工

为了实现数控程序的 DNC 加工，首先需调用函数 `cnc_dncstart2(h,fname)`，程序名 `fname` 定义为 16 个字节的字符串变量，在运行过程中由 `cnc` 动态分配。函数执行后，检查返回值，如为 `EW_OK`，说明 DNC 操作已开始，此时返回的程序号为 `NULL`。接着调用函数 `cnc_dnc2(h,&n,prg)`，输出数控程序进行 DNC 操作，它有 3 个参数，其中：`h` 为通信句柄，`n` 为将要输出的字符数，`prg` 为指向存储程序区域的指针。同样需检查返回值，直至 DNC 操作完成，最后调用函数 `cnc_dncend2(h,DNC_NORMAL)` 结束 DNC 操作。图 2 为 CL-20A 数控车床 DNC 模块的开发流程。

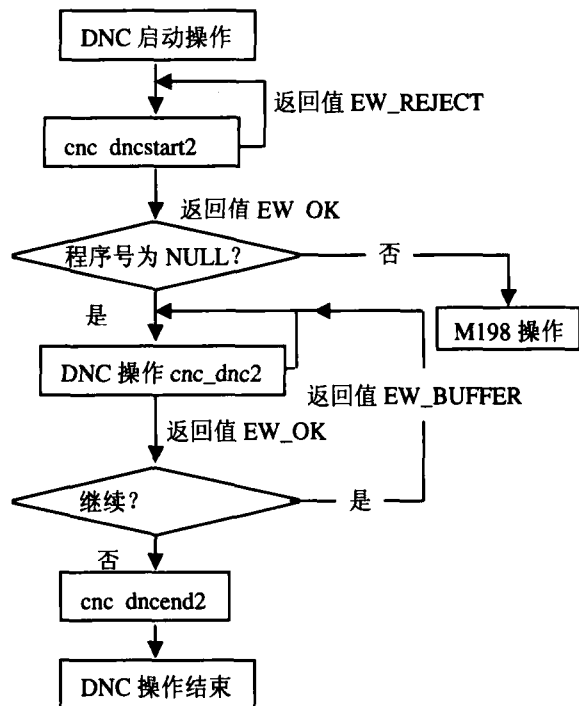


图 2 机床 DNC 编程流程图

为了能够进行正常的 DNC 操作,除了按图 2 流程编程外,还需进行如下操作:①将机床的模式设置为 DNCI 模式(TAPE 模式),此时 G043.5 信号为 1;②将#20 参数设为 6,选择基于以太网的通信方式;③将参数 8706 的第 1 位设为 1,以保证微机与机床的正常通信。

3 PLC 控制模块的开发

为了体现生产线的柔性效率,数控加工中的上下料由机械手(YASNAC XRC SK16X)完成,但必须协调好机械手与机床的动作。为此,添加了一块外置 fp0-PLC,以实现两者间的通信,梯形图程序如图 3。

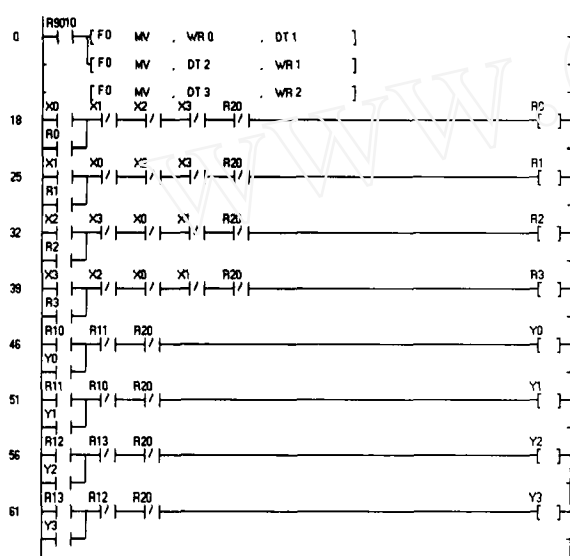


图 3 PLC 控制模块梯形图程序

机械手的输出信号(即:PLC 的输入信号)包括:关门(X0)、开门(X1)、松开卡盘(X2)、夹紧卡盘(X3)4 个动作,机械手的输入信号(即:PLC 的输出信号)包括:门关(Y1)、门开(Y0)、卡盘松开(Y2)、卡盘夹紧(Y3)4 个状态。在机械手示教程序中,输出控制信号,例如:输出开门信号,使 X1 输入继电器常开触点通(常闭触点断),则内部继电器 R1 线圈通电,由于互锁关系,R0, R2, R3 线圈不可能通电,故 WR0 的值为 2,同时将 DT1 的值赋 2。而 PLC 控制模块可实时读取数据寄存器 DT1 的值,并根据值的不同而执行不同的功能,例如:DT1=2 时,调用函数 Open_Door()进行开门控制。门开到位后,调用函数 OnDoorOpenState(),使 DT2 为 1,同时将 WR1 的值赋 1(即:R10 为 1, R11, R12 等为 0),则 Y0 线圈通电,从而机械手可以检测到机床门处于开的状态。微机与 PLC 的串行通讯程序的开发详见文献[3],其余功能的实现与此相似,限于篇幅,不再赘述。

参考文献:

- [1] 佚名. Orbix Programmer's Guide[Z]. IONA Technologies PLC, 1997.
- [2] 佚名. FANUC OPEN CNC FOCAS1/2 Libraries Edition 02.4[R]. FANUC LTD, 2002.
- [3] 蔡锦达, 申屠里峰, 王德福. 用 VC++ 开发微机与松下 PLC 间的通讯程序[EB/OL]. <http://www.ca800.com/maga/make/plc/detail.asp?id=397>, 2003-12.

Development of DNC Module for CNC Machine Tools Based on CORBA

ZHANG Aihong

(Department of Mechanical and Electrical Engineering, Wuxi College of Technology, Wuxi, Jiangsu 214121, China)

Abstracts: Real time control of CNC machine from computer is indispensable for flexible manufacturing system. Wuxi College of Technology Network DNC Module for CNC machine is one of the important modules in FMS II(Flexible Manufacture System), which can communicate with other modules based on CORBA, and consists of three sub-modules, including communication module, DNC module and PLC module. This paper mainly describes the development of these modules and some samples are provided as well.

Key words: CORBA; communication; DNC; control

(责任编辑: 王璐)