文章编号：1672-2892(2010)02-0231-04

PROFINET CBA 组件的应用实现

张学东，谢兴全，李　潮

(西南计算中心，四川 绵阳 621900)

摘　要：PROFINET 是基于工业以太网的开放标准，PROFINET 基于组件的自动化(CBA)技术通过采用标准自动化组件实现了分布式自动化系统简单高效的互联。本文研究了 PROFINET CBA 的通信原理、模型，基于 PI 组织的 PROFINET CBA Runtime Source 代码开发了 PROFINET CBA 自动化组件，实现了 PROFINET CBA 组件之间的实时通信，对于构建基于 PROFINET CBA 的分布式通信系统具有一定的工程意义。

关键词：RPOFINET 基于组件的自动化技术；实时以太网；分布式组件对象模型；iMap 工程

中图分类号：TN915.04；TP273　　**文献标识码**：A

Implementation of PROFINET CBA component

ZHANG Xue-dong，XIE Xing-quan，LI Chao

(South-west computing center，Mianyang Sichuan 621900，China)

Abstract：PROFINET is the open standard for industrial Ethernet. PROFINET Component Based Automation(CBA) technology makes the communication between distributed automation systems easier and more efficient by providing standardized automation components. This study introduced the theory and model of PROFINET CBA. PROFINET CBA components based on PROFINET Runtime Source were developed and a real-time communication system was implemented. PROFINET CBA technology can be applied in constructing a distributed automation communication system.

Key words：RPOFINET Component Based Automation；real-time Ethernet；Distributed Component Object Model(DCOM)；iMap

PROFINET 是一个基于工业以太网的开放自动化标准，其重要突破是基于组件的自动化(CBA)技术[1]，通过将不同的控制系统打包为标准组件，在组件中使用统一的通信接口，CBA 技术很好地降低了由于当前工厂中多供应商、多种控制平台并存所带来的系统复杂性。传统的自动化工程方法(面向连接的方法)中，通信作为控制和组态的一部分，每一次使用设备模块时，都需要对控制器进行组态和调试。分布式控制器和技术模块独立运行(面向对象的方法)带来的好处是：控制器程序单独设计，在控制系统最后装配时，采用 PROFINET 工程工具实现，提高了系统的开发和调试效率，同时提高了系统模块的重复利用率。组件化技术作为 PROFINET CBA 的核心技术，对于实现控制系统 CBA 封装，构建基于 PROFINET CBA 的工程系统具有实际的工程应用价值。

1　PROFINET CBA 技术概述

PROFINET CBA 基于 IEC61499-1(用于分布式工业过程测量与控制系统功能块的标准)定义了一种跨制造商的通信和工程模型，用于实现技术功能模块之间系统级(包含控制器之间)的通信。PROFINET CBA 除了实现分布式组件对象模型(DCOM)通信外，由于采用了优化的软实时通信通道，

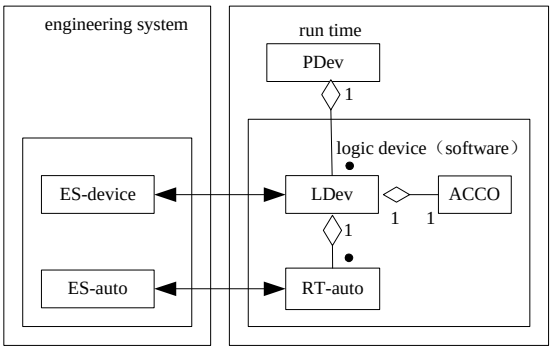


Fig.1 Component model of PROFINET CBA
图 1 PROFINET CBA 组件模型

收稿日期：2009-10-22；**修回日期**：2009-11-18

基金项目：中国工程物理研究院科技发展基金资助项目(09–0941)

能够对数据链路的数据帧直接控制，实现基于实时时钟芯片 1(Real Time Class1，RTC1)的实时通信，实时性在 10 ms~100 ms 之间[2]。PROFINET CBA 的组件模型见图 1[3]。

PROFINET CBA 运行模型定义了组件的运行对象，物理设备对象 PDev(PhysicalDevice)代表设备的硬件；逻辑设备对象 LDev(LogicalDevice)是设备的软件表示；运行时自动化对象 RT-Auto(Runtime Automation)是具有相应数据区的可执行程序，为 PROFINET 控制器提供实际的技术功能，在 RT-Auto 中定义了应用代码的接口函数及接口数据类型；主动控制连接对象(Active Control Connection Object，ACCO)是互连的中央管理和操作点，通过内部接口访问自身 RT-Auto 的数据，并将数据传输到通信伙伴的 ACCO。

PROFINET CBA 的工程模型中定义了组件的 XML 描述文件[4]，PROFINET 工程工具通过组件的 XML 描述文件调用 PROFINET CBA 组件的接口，必须和组件的运行模型接口一致。

2 PROFINET CBA 组件运行对象的实现

PROFINET Runtime Source 是一种与操作系统无关的 PROFINET Runtime 模型的软件实现，以源代码的形式通过 PI 组织网站获得，提供了通信模型的底层通信函数调用、应用模型对象的基础类[5]。基于 PROFINET Runtime Source 代码实现 PROFINET CBA 自动计数器组件，自动计数器 PROFINET CBA 组件的构成见图 2。

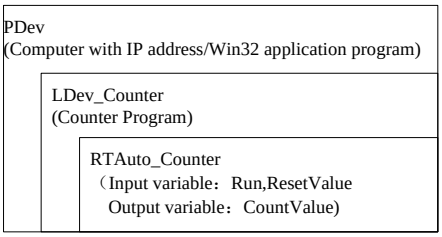


Fig.2 Construction of Auto-Counter PROFINET CBA component
图 2 自动计数器组件的构成

PDev 物理设备由计算机上 PROFINET Runtime Source 下的 Phydeb.exe 应用程序实现，运行时需要在 DCOM 服务端进行注册。LDev_Counter 和 RTAuto_Counter 具体的开发流程包括：组件接口定义(Interface Definition Language，IDL)文件的编辑；通过编译程序(amashcv.exe midl.exe)对 IDL 文件编译产生调用库文件；构建 LDev_Counter 和 RTAuto_Counter。

2.1 自动计数器组件接口定义 IDL 文件

在自动计数器组件的接口定义 IDL 文件中，RTAuto_CounterClass 类的接口定义如下：

```
coclass RTAuto_CounterClass
{
    [default] interface ICounter;          //用户自定义接口
    interface ICBARTAuto;                  //标准接口
    interface ICBARTAuto2;                 //标准接口
    interface ICBABrowse;                  //标准接口
    interface ICBABrowse2;                 //标准接口
}
```

其中自定义接口界面 Interface Icounter 的输入输出接口变量定义如下：

```
interface ICounter : IDispatch
{
    [propget] HRESULT CounterValue([out, retval] long *puVal);
    [propput] HRESULT Run([in] char bVal);
    [propput] HRESULT ResetValue([in] long bVal);
}
```

接口界面定义了计数器组件的 Run,Counter Value,Reset Value 输入输出变量，用于启动、复位计数器的值。

2.2 构建 LDev_Counter 和 RTAuto_Counter

LDev_Counter 和 RTAuto_Counter 基于 PROFINET Runtime Source 代码编程实现。

a) LDev_Counter 和 RTAuto_Counter 的构建流程

LDev_Counter 和 RTAuto_Counter 的构建包括添加 LDev 对象、构建 RTAuto 对象、添加 RTAuto 对象、注册 RTAuto 对象、注册 LDev 对象，流程见图 3。

b) RTAuto_Counter 的接口变量初始化

在 CBA_RTAuto_Construct()中，构建并初始化 RTAuto_Counter，实现对 RTAuto Interface-Table 和 RTAuto Property-Table 的操作。RTAuto Interface-Table 中定义了接口数据的方法对象，RTAuto Property-Table 定义了接口的数据对象。访问接口数据对象，将其映射为全局变量，进行初始化，过程如下：

Fig.3 Implementation flow of generating LDev_Counter and RTAuto_Counter

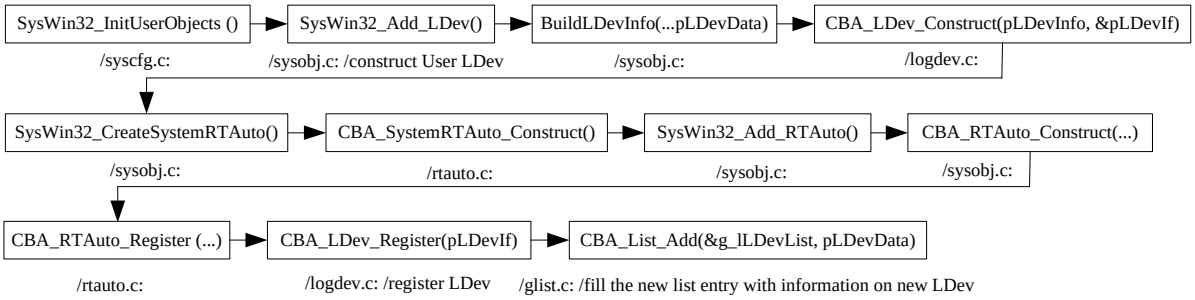


图3 LDev_Counter和RTAuto_Counter的实现流程

```
g_pPropertyCounterValue = (SysWin32_property_t *) pCbaPropTbl->hUser;//获得 CounterValue 的接口对象指针
g_pPropertyRun = (SysWin32_property_t *) pCbaPropTbl->hUser;          //获得 Run 的接口对象指针
* (char*)g_pPropertyRun->pPropData = 1;                              //初始化*g_pPropertyRun 值为 1,运行计数器
g_pPropertyReset = (SysWin32_property_t *) pCbaPropTbl->hUser;        //获得 CounterValue 的接口对象指针
* (long*)g_pPropertyReset->pPropData = 50;                           //初始化* g_pPropertyReset 值为 50，计数为达到 50 时，自动复位 0
```

c) RTAuto_Counter 的计数功能的实现

在系统定时器的 1 s 定时函数调用中更新 RTAuto_Counter 的接口数据，调用函数如下：

```
void CBA_FCT_HUGE SysWin32_TimerCallBack(
if ((g_pPropertyCounterValue != NULL) && (g_pPropertyRun != NULL))
    {//判断输入是否有效
    if (* (char *)g_pPropertyRun->pPropData)
        {(* (long*)g_pPropertyCounterValue->pPropData)++;
        }//如果 Run 为 1，计数值加 1
    if(*(long*)g_pPropertyCounterValue->pPropData>*(long*)g_pPropertyReset->pPropData )
        {* (long*)g_pPropertyCounterValue->pPropData = 0;
        }//如果计数值*pPropertyCounterValue 超过设定值 50，自动复位为 0
    }
```

3　PROFINET CBA 组件 XML 描述文件的编辑

PROFINET CBA 组件的 XML 描述文件定义了组件的工程应用接口，用于在 PROFINET CBA 工程工具中对组件进行调用。PROFINET CBA 组件的 XML 描述文件的组织架构见图 4。

ComponentMaster 定义了组件的节点信息、物理设备、逻辑设置的信息。

组件的 XML 描述文件需要根据 PROFINET CBA 规范进行修改，需要定义的部分主要包括：网络节点的 IP 地址和服务端的 IP 地址一致；逻辑设备的 Component ID 和 Rev 与运行的组件一致；Function 名字需要和 RTAuto 的名字 RTAuto_Counter 一致。

iMap 工程环境中调用 PROFINET CBA 组件时，读取组件信息和 XML 文件中的信息进行比较诊断，一致的情况下才能进行组件的通信连接。

4　PROFINET CBA 组件的测试

组件运行时，需要在 PROFINET CBA 服务端和客户端进行注册。在 PROFINET CBA 服务端注册、设置并运行 PDev 服务。客户端注册 proxy/stub：regsver32 profinetrtps.dll，同时配置 DCOM 的访问权限。采用 PNTest 工具对 PROFINET CBA 组件进行接口测试，LDev 的测试信息见图 5。在 iMap 工程工具中调入分布在 2 台 PC 机上的计数器、运算器组件，进行通信连接和测试，见图 6。

在组件之间通信数据量为 4 个字，通信周期为 10 ms 和 50 ms 情况下，对通信状态进行在线监测，组件之间通信正常。基于 PROFINET CBA Runtime Source 代码开发的 PROFINET CBA 组件能够满足一般工厂自动化 10 ms~100 ms 的实时性能要求。

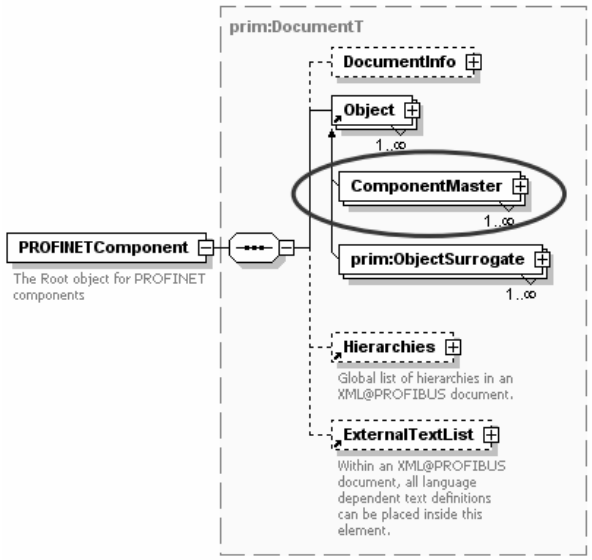


Fig.4 XML file structure of PROFINET CBA component
图 4 PROFINET CBA 组件的 XML 文件结构

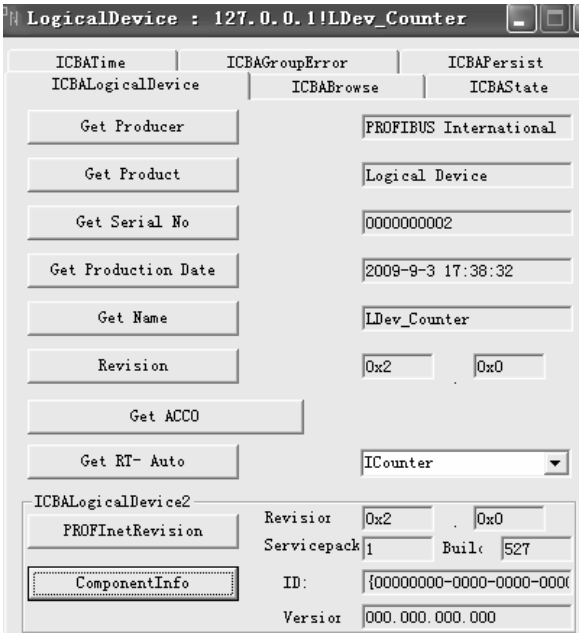


Fig.5 LDev test of PROFINET CBA component
图 5 PROFINET CBA 组件的 LDev 测试

5 结论

PROFINET CBA 改变了传统的工程设计方法，通过将控制系统组件化，使得控制和通信独立开发，同时组件化的功能模块可以复用，开发效率得以提高。基于PROFINET CBA Runtime Source代码实现的 PROFINET CBA 自动化组件，对于实现控制系统的 PROFINET CBA 封装，构建基于 PROFINET CBA 的分布式自动化通信系统有实际的工程应用价值。

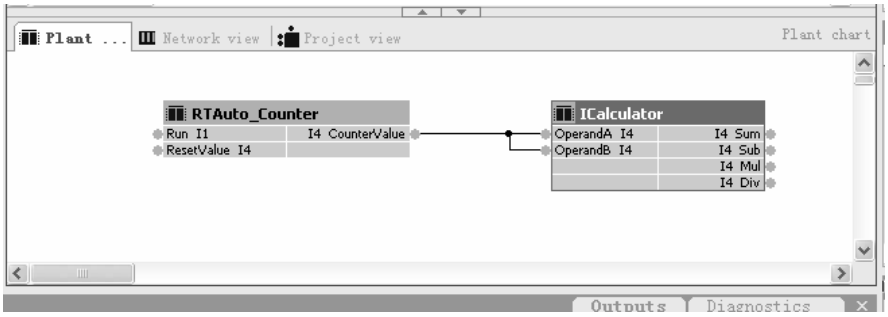
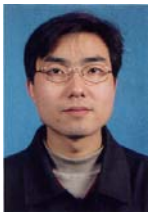


Fig.6 Linking test of CBA component in iMap software
图 6 PROFINET CBA 组件在 iMap 中的连接测试

参考文献：

[1] PROFIBUS International. PROFINET Technology and Application System Description[Z]. 2006.
[2] 张广法,唐钟,谢阅. 基于 PROFINET 的网络通信系统[J]. 信息与电子工程, 2009,7(2):164-167.
[3] Raimond Pigan,Mark Mette. 西门子 PROFINET 工业通信指南[M]. 北京:人民邮电出版社, 2007.
[4] PROFIBUS International. PROFINET CBA Architecture Description and Specification Version 2.20[Z]. 2008.
[5] PROFIBUS International. PROFINET CBA Implementation Guide Version 2.0[Z]. 2004.

作者简介：



张学东(1976-)，男，河南省南阳市人，硕士，工程师，研究方向为工业通信、工业控制、嵌入式控制.email:sidafaming@126.com.

谢兴全(1972-)，男，四川省绵阳市人，学士，高级工程师，研究方向为计算机应用、工业控制.

李 潮(1976-)，男，河南省南阳市人，学士，工程师，研究方向为计算机应用、工业控制.