

电压无功控制系统特殊进程配合类设计

黄 凯,顾 全,叶清华

(南京南瑞继保电气有限公司,江苏 南京 211100)

摘要:对于一体化的 EMS 系统,无功控制策略优化进程和控制服务进程的交互无需通过代价高昂的网络编程实现,它们可以通过共享实时库交互,这必将涉及进程同步和共享内存保护问题,传统的互斥及信号灯不能满足无功控制系统的特殊要求。以自适应通信环境(ACE)中间件为基础,开发了进程同步信号灯类,并提出了无功控制中的进程交互方案解决无功控制系统的特殊要求,该方案还可应用于调度员培训仿真系统(DTS)中保护进程和主培训仿真进程的配合,试验证明该方法正确、有效。

关键词:电压无功控制; 进程同步; 自适应通信环境

中图分类号: TM 715

文献标识码: B

文章编号: 1006-6047(2006)02-0057-05

1 问题的提出

电压无功控制对于电力系统安全经济运行的作用越来越得到重视,对于无功控制的策略、算法的研究很多^[1-7],但对底层实现机制论述较少。

随着电压无功控制系统的实用化以及对不同厂家软件兼容性的要求越来越高,电压无功控制系统除了需将自身的控制策略以命令报文的形式下发给 SCADA 或集控站执行,还需要过滤第 3 方控制系统的电压无功控制命令。传统的方法中电压无功控制策略进程与控制服务进程为同一进程,无论何时都打开 Socket 端口进行侦听,当用户决策采用第 3 方控制时便转发第 3 方报文,否则运行无功控制主程序,这样的处理方式一方面带来程序执行效率问题,另一方面引发进程可靠性问题,即,无论是处理第 3 方命令出错或电压无功控制决策进程内部发生错误都将导致

2 种服务都不可用。因此,合理的方式是将无功控制决策模块和无功控制执行模块分开,这样,一种服务故障时并不影响另一服务的运行。

当无功控制决策模块和无功控制执行模块分为 2 个进程进行时,两者需进行交互,交互内容主要是前者将无功控制命令传递给后者,而后者需将命令的执行结果返回给前者。要完成这种交互工作,可以采用 2 种方式:一种是采用网络编程模式,这需要无功控制策略进程将控制命令组包,并发送给无功控制执行进程,而无功控制执行进程需要解包校验,随后将命令执行结果组包,返回给无功控制策略进程,无功控制策略进程在收到执行结果时再决策自己下一步的控制策略;另一种是采用共享内存方式,无功控制策略进程负责将需执行的命令写入共享内存,无功控制执行进程从共享内存中读出命令并执行,在执行完后将执行结果写回共享内存。可见:后一方式比前一方式省去组包、解包和报文传输过程,因而效率较高,对于不同进程可以共享同一实时库的系统,

收稿日期:2005-05-11;修回日期:2005-10-26

采用后一方式有明显的优越性,但是要体现这种优越性必须要解决 2 个进程的特殊同步问题。

由于操作系统调度时,进程或线程的上下文切换是随机的,对访问同一内容的进程或线程必须进行同步,对临界区代码必须进行保护,这可以通过互斥量或信号灯解决。但是,本文的特殊之处在于调度自动化系统的 4 个特殊要求。

a. 电压无功控制系统作为电力系统调度自动化软件,要求尽可能为电力系统提供不间断服务,即电压无功控制决策进程和电压无功控制执行进程的任一方中断运行(正常/异常)不应影响另一方运行。具体包括:一方进程退出时,另一方进程不得死锁;一方进程退出时,另一方进程不应为获取锁降低效率;退出运行的进程重新投入运行时,2 个进程的互锁机制自动恢复正常运行。

b. 各进程获取锁不应随机产生,在条件满足时进程应自动获得锁。

c. 当所有进程退出时信号灯应自动销毁,以免影响下次进程启动时,因信号灯的初始值而影响锁的正确获取与释放。

d. 系统要求具备跨平台特性,要求可在 Windows, Unix, Linux 等多种操作系统上运行。

要同时解决上述要求,需要进行特殊编程,本文就此提出了解决方案。

2 解决方案所依赖工具与平台

2.1 自适应通信环境简介^[8]

本文的解决方案需要借助于自适应通信环境 ACE(Adaptive Communication Environment)。

2.1.1 ACE 概述

ACE 是一个面向对象框架(framework),在这个框架中,提供了一些能用于开发并发通信软件的模式。ACE 的目标是开发高性能和实时性的通信服务应用软件。通过使用 ACE 提供的模式,可以大大简化涉及进程间通信、事件分解、分发和并发的面向对象应用软件的开发过程。为分离关系、降低复杂度、允许功能模块划分,ACE 被设计为层次化体系结构^[9],其基础是操作系统适配层(os adaptation layer)和 C++ 封装层(C++ wrapper facades layer)^[10]。

本文仅就用到的操作系统适配层和 C++ 封装层作简要介绍。

2.1.2 操作系统适配层

操作系统适配层封装了原始的、基于 C 的 OS API,屏蔽了各类操作系统间 API 的差异,向运用 ACE 的软件开发人员提供统一的 API 接口。该层将保护 ACE 的高层不依赖于平台,从而保证用 ACE 编写的代码与平台的独立性。该层在它所支持的任意操作系统环境下工作几乎完全相同,这样,开发人员花费很少的精力或根本无需花费精力,就可以将一个 ACE 应用从一个平台移植到另一个平台。

目前,操作系统适配层所支持的操作系统有:实

时操作系统(VxWorks, Chorus, LynxOS, RTEMS, OS/9, QNX Neutrion, pSoS),大多数 UNIX(SunOS, SGI IRIX, HP_UX, Tru 64, AIX, DG/UX, Linux, SCO, Unixware, NetBSD, FreeBSD),Win 32(Win NT, Win 95, WinCE),MVS OpenEdition, Cray UNICOS。

该层提供的主要功能有并发同步、进程间通信、事件分解机制、动态连接、文件系统管理和 c 库函数。

2.1.3 C++ 封装层

C++ 封装层,又称为 C++ 包装层,包括可用于创建高可移植性和类型安全的 C++ 应用的 C++ 类。这是 ACE 工具包中最大的一部分,大概包含了全部源代码的 50%。该层通过强化和封装操作系统适配层的接口,提供了类型安全的 C++ 接口^[10]。软件开发人员可通过它提供的类与对象,以继承、繁殖、例化的方式组合这些可重用的模块,以此开发出应用软件。在 ACE 实现细节中,低一层的操作系统适配层的接口被映射和抽象成类/对象。

C++ 封装层提供和操作系统适配层大致相同的功能,因 C++ 是强类型语言,使用该层的类比原先更安全。由于抽象和封装,为方法的执行增加了开销,实际上,ACE 通过内联方法减小方法调用的开销。

2.2 系统平台简介

RCS-9001 SYSMAN 是 RCS-9001 能量管理系统的一部分,是一个基于多现场、可远程管理的系统管理平台。该平台是一个基于分布式管理框架的管理子系统。从简单的由 1~2 个子系统组成的小型系统,到复杂的跨电力企业各个自动化和管理、决策系统的基于多现场的大型系统,RCS-9001 SYSMAN 系统管理平台可用于对这些系统中各类硬件资源的使用情况进行监视;对各软件部件的运行状态进行监视、记录和控制,以改变其运行的状态;并对系统的安全性、性能、日志等进行统一管理。

RCS-9001 SYSMAN 是在复杂的 UNIX, LINUX 和 Windows 异构环境中,对 RCS-9001 系统的用户、软件和硬件资源进行集中管理的系统管理平台。它为 RCS-9001 EMS/DMS/TMS 系统管理员和开发人员轻松管理系统软硬件资源、及时监视系统运行状况、分析并找出系统已发生或即将发生的问题,并对系统运行进行相应调整和控制提供完整的解决方案。

解决本文的问题时,需要通过系统管理平台得到另一进程的状态,通过调用类 `sophic_net_api` 的 `sys_get_pid(char * processname)` 方法,可以返回指定名称的进程状态。

3 进程同步类设计与实现

3.1 进程同步类的实现代码

为解决本文开头提出的问题,设计了如下的类:

```
class SN_InterProcess_Semaphore
{
    Class SN_Improved_Semaphore:
```

```

public ACE_Semaphore
{
public :
SN_Improved_Semaphore (ULong count=1,Long type=
USYNC_THREAD,const Char*name=0,void *arg = 0,
Long max = 0x7ffffff)
:ACE_Semaphore(count,type,name,arg,max){};
};
public:
//构造函数
SN_InterProcess_Semaphore (const Char * name,Long
init_number)
# if defined ACE_WIN32
:lock_ (init_number,USYNC_PROCESS,name){}
# else
{ lock_.open ( name , ACE_SV_Semaphore_Simple::
ACE_CREATE,init_number,1,0666);}
# endif /*ACE_WIN32*/
~SN_InterProcess_Semaphore()
{
# ifndef ACE_WIN32
lock_.close();
# endif
}
Long acquire()
{
# if ! defined ACE_WIN32
return lock_.acquire(0,SEM_UNDO);
# else
return lock_.acquire();
# endif
}
//每隔 10 ms 获取信号灯一次,超时返回-1
Long acquire(Long mseconds)
{
Long ret,i;
ACE_Time_Value tv(0,10000);
Long count = mseconds / 10;
for(i = 0;i < count;i++)
{
ret = try_acquire();
if(ret == 0) return 0;
ACE_OS::sleep(tv);
}
return -1;
}
Long acquire(Double seconds)
{
Long mseconds = (Long)(seconds * 1000);
return acquire(mseconds);
}

```

```

Long try_acquire()
{
# if ! defined ACE_WIN32
return lock_.tryacquire(0,SEM_UNDO);
# else
return lock_.tryacquire();
# endif
}
Long release()
{
# if ! defined ACE_WIN32
return lock_.release(0,SEM_UNDO);
# else
return lock_.release();
# endif }
private:
# if defined ACE_WIN32
SN_Improved_Semaphore lock_;
# else
ACE_SV_Semaphore_Complex lock_;
# endif /*ACE_WIN32*/
};

```

3.2 类设计说明

由于本文的设计要求一个进程的退出不影响另一个进程,即当一个进程如果在获得信号灯期间因异常而中止,另一个进程不能死锁;此外,当所有使用信号灯的进程退出时,信号灯应被自动销毁。

本类设计的核心在于使用了 ACE 的 2 个类: ACE_Semaphore 和 ACE_SV_Semaphore_Complex,对这 2 个类分别通过预编译选项 ACE_WIN32 在 Windows 和 Unix 系统上分别使用。

ACE_Semaphore 类在 Windows 平台封装的是 Windows 操作系统提供的 API 函数 CreateSemaphore, ReleaseSemaphore, WaitForSingleObject, WaitForMultipleObjects 等。由于 Windows 平台具有句柄计数功能,当该类的析构器会销毁本信号灯时,系统只是将使用句柄的个数减一,并不真正销毁,因此只有当使用线程或进程都退出运行时,信号灯的句柄引用计数降为 0 时,系统才会将信号灯销毁,而该类在 Unix 并不存在上述管理,析构器会将信号灯删除而不管别的进程是否使用该信号灯,这可能使别的进程死锁,因此在 Unix 系统上使用 ACE_SV_Semaphore_Complex 类。

类 ACE_SV_Semaphore_Complex^①是一个很复杂的信号灯,它在初始化或关闭信号灯时可以正确处理相关的竞态条件,它利用 System V(目前大多数 Unix 系统都支持的一种进程通信机制)的调整特

① SCHMIDT D C.,ACE 源代码开放软件包 5.2 版(SV_Semaphore_Complex.h,SV_Semaphore_Complex.cpp,SV_Semaphore_Complex.i),2001.(Douglas C. Schmidt,ACE Version 5.2(SV_Semaphore_Complex.h,SV_Semaphore_Complex.cpp,SV_Semaphore_Complex.i),2001).

性,对进程每次打开或关闭信号灯进行计数,即使程序非正常退出时也会被计数。从本文设计的 SN_InterProcess_Semaphore 类可见:类析构时调用的是 close 方法,该方法只有在本进程是使用信号灯的最后一个进程时才删除信号灯。

由本节可见:利用 ACE 仅仅用了数十行代码便开发出了具有复杂机制、具备设计功能且具有跨平台特性的进程同步信号灯类。利用该类和系统平台提供的功能,通过适当的流程设计,可以解决本文开头提出的问题。此外,本文设计的进程同步类还提供了超时返回功能,可以让进程在指定时间内获取不到信号灯时不至于死锁。

4 电压无功控制系统同步架构

电压无功控制策略进程和电压无功控制服务进程需交替进行,如图 1 所示。

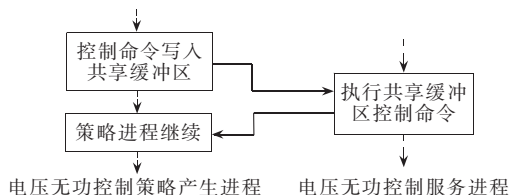


图 1 进程交替示意图

Fig. 1 Interaction of AVC processes

图 1 显示,电压无功控制策略产生进程在产生控制命令后将控制命令写入共享缓冲区,等控制服务进程将控制命令下发并返回结果后,电压无功控制策略产生进程根据命令的执行情况(这里的执行情况不是指设备实际动作情况,而是指命令是否成功下发到厂站设备。实际的设备调节情况要通过采集系统获得,而不能仅根据报文返回情况判断)进行后续决策。

上述流程需要 2 个进程交替进行,仅靠 1 个信号灯是不够的,因此本文设计 2 个信号灯来保证 2 个进程的交替进行,同时保证一个进程退出后另一个进程不会死锁,还保证了当退出的进程重新投入运行后信号灯能正常工作。程序流程改变见图 2。

流程设计说明如下。

a. 2 个进程都执行信号灯 A、B 的创建动作,但是该创建动作当信号灯已存在时实际执行的是打开动作,这保证任一进程退出运行后再次投入时还可以和另外一个进程共享同一信号灯。此外,第 1 个启动的进程会以 0 初始化信号灯,这保证 A、B 信号灯的資源在被释放前处于不可用状态。信号灯 A 负责控制策略进程的阻塞,信号灯 B 负责控制服务进程的阻塞。

b. 调用系统平台工具,获取另一进程状态,控制策略进程中仅当控制服务进程存在时才释放信号灯 B,由于信号灯 B 仅在控制策略进程中释放,所以,只有当控制策略进程释放信号灯 B 时,控制服务进程才可以执行共享缓冲区中的命令。应注意到获取信

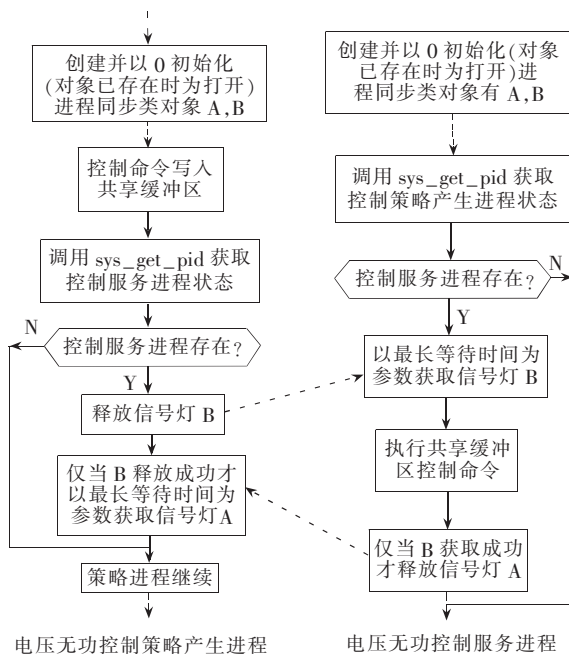


图 2 无功控制进程同步流程

Fig. 2 Flowchart of var control process synchronization

号灯是以最长等待时间来获取信号灯 B 的,即获取信号灯 B 不成功时控制服务进程依然会执行共享缓冲区的命令。这么做的目的是:获取信号灯 B 的最长等待时间其实是电压无功控制策略产生进程的正常执行周期时间,如果在这么长的时间内不能获得信号灯 B 意味着策略产生进程已异常退出,因此需要将共享缓冲区中还没执行完的命令执行完毕。实际上当共享缓冲区的命令全部执行完毕后该模块不再起作用,因此流程不会有問題。

c. 策略产生进程仅当信号灯 B 释放成功时才会以最长等待时间获取信号灯 A,因为如果 B 不能释放成功则控制服务进程就不可能释放信号灯 A,此时控制服务进程可能已出现异常,因此等待一个最长时间后策略进程恢复运行,如果此时控制服务进程退出运行后,下个周期时策略进程捕捉到控制服务进程退出则不会再进行信号灯操作,策略进程正常进行。

d. 在控制服务进程中,仅当获取信号灯 B 后才释放 A 的原因与 c 类似,它保证策略进程异常退出后控制服务进程最多等待一个最长周期便可正常运行。

上述设计已在 SunOS 5.8 操作系统环境和 Windows 2000 系统上测试通过。试验证明本文方案可以解决本文开头提出的问题,读者也可以在 ACE 环境下测试,测试时因没有本公司的 RCS-9001 SYSMAN 系统平台可以采用变通方法获取另一进程的进程状态,然后采用本文的设计进行试验。

5 推广应用

本文的进程同步类不局限于电压无功控制系统,其他有类似需求涉及 2 个进程配合问题时都可以借助本文提供的类和方法。该方法在 RCS-9001 能量管理系统中还有一个应用,就是在调度员培训

仿真中,由于动态潮流(包括频率计算)主进程和保护仿真进程在进行故障处理时也有类似配合,即主流程中的事件管理模块在触发系统故障时需要唤醒保护仿真进程,待保护仿真进程将故障处理完毕后,将处理结果返回时,主流程将处理结果反映到系统中后重新进行潮流计算,这2个进程也都要求一方因异常退出运行后另一方需正常运行,而当异常进程重新启动后两者还能正常配合,应用表明本文设计的进程同步类正确地实现了设计目标。

ACE是高度可移植的开放源码主机基础设施中间件软件包,从根本上改变了网络化应用和中间件在许多操作系统上的设计和实现方式,从大型的“财富550强”到小型的新兴企业,再到高校和工业试验的高级研究项目,数千开发团体正在使用ACE^[3]。其设计及实现上使用了众多前沿的计算机软件技术,对于缩短开发周期、提高开发效率有着及其重要的意义。ACE工具和源码可以在<http://ace.ece.uci.edu>或<http://www.riverace.com>上自由获取。

参考文献:

- [1] 胡伟,卢强.混成电力控制系统及其应用[J]. 电工技术学报,2005,20(2):11-16.
HU Wei,LU Qiang. Hybrid power control system and its application[J]. **Transactions of China Electrotechnical Society**,2005,20(2):11-16.
- [2] 丁晓群,黄伟,张文俊,等.基于电压控制区的主导节点电压校正方法[J]. 电网技术,2004,28(14):44-48.
DING Xiao-qun,HUANG Wei,ZHANG Wen-jun,et al. A pilot bus voltage correction method based on voltage control area[J]. **Power System Technology**,2004,28(14):44-48.
- [3] 孙宏斌,张伯明,郭庆来,等.基于软分区的全局电压优化控制系统设计[J]. 电力系统自动化,2003,27(8):16-20.
SUN Hong-bin,ZHANG Bo-ming,GUO Qing-lai,et al. Design for global optimal voltage control system based on soft identification of secondary control zone[J]. **Automation of Electric Power Systems**,2003,27(8):16-20.
- [4] 朱军飞,唐寅生,周全仁.电网AVC分层控制[J]. 电力设备,2002,3(4):28-30.
ZHU Jun-fei,TANG Yin-sheng,ZHOU Quan-ren. Hierarchical control structure of AVC[J]. **Electrical Equipment**,2002,3(4):28-30.
- [5] 丁晓群,陈晟,许杏桃,等.全网无功电压优化集中控制系统在泰州电网的应用[J]. 电网技术,2000,24(12):21-23.
DING Xiao-qun,CHEN Sheng,XU Xing-tao,et al. Application of optimization and centralized control system for reactive power/voltage in Taizhou power network[J]. **Power System Technology**,2000,24(12):21-23.
- [6] 孙宏斌,吴文传,张伯明,等.安全约束下的全局最优无功控制的仿真研究[J]. 电力系统自动化,1999,23(5):4-7.
SUN Hong-bin,WU Wen-chuan,ZHANG Bo-ming,et al. Study on simulation of security considered global reactive optimal control[J]. **Automation of Electric Power Systems**,1999,23(5):4-7.
- [7] 许文超,郭伟,李海峰,等.AVC应用于江苏电网的研究[J]. 继电器,2003,31(5):23-26.
XU Wen-chao,GUO Wei,LI Hai-feng,et al. Preliminary study on automatic voltage control of the electric network in Jiangsu province[J]. **Relay**,2003,31(5):23-26.
- [8] SCHMIDT D C,HUSTON S D. C++ 网络编程[M]. 於春景,译. 武汉:华中科技大学出版社,2003.
- [9] BUSCHMANN F,MEUNIER R,ROHNERT H,et al. Pattern-oriented software architecture—a system of patterns[M]. New York:Wiley and Sons,1996.
- [10] SCHMIDT D C,HUSTON S D. C++ 网络编程[M]. 马维达,译. 北京:电子工业出版社,2004.

(责任编辑:柏英武)

作者简介:

黄凯(1978-),男,安徽来安人,工程师,硕士,从事调度自动化系统的设计与研究工作(E-mail:huangkai@rcs-9000.com);

顾全(1970-),男,江苏高邮人,高级工程师,硕士,从事调度自动化系统的设计与研究工作(E-mail:guquan@rcs-9000.com);

叶清华(1977-),女,江苏南京人,工程师,硕士,从事调度自动化系统的设计与研究工作(E-mail:yeqh@rcs-9000.com)。

Special class design of interprocess synchronization in voltage and var control system

HUANG Kai, GU Quan, YE Qing-hua

(NARI-Relays Electric Co., Ltd., Nanjing 211100, China)

Abstract: For an integrative energy management system, shared memory can be used for interaction between decision-making process and control-server process instead of expensive network programming. However, it needs process synchronization and shared memory protection, which can't be met with traditional mutex and signal lamp. A process synchronization signal class is developed based on adaptive communication, and a scheme for process interaction in var control is presented, which is also suitable for interaction between protection process and main training process in dispatcher training simulator. Tests verify its correctness and effectiveness.

Key words: voltage and var control; process synchronization; adaptive communication environment