基于上位机的混合流程顺序控制系统①

翁 元

(国电南京自动化股份有限公司, 南京 210003)

摘 要: 以计算机为基础的生产过程控制与调度自动化系统, 常会遇到包含并发控制流程、多条件多路径的复杂控制过程等等. 基于当前计算机和通信技术的发展水平, 该文提供了一种通过在调度中心上位机运用全组态配置和后台程序搭配的设计方法, 实现混合流程的顺序控制, 可以有效缩短控制时间、减少误操作从而能使下位机运行的可靠性明显提高, 具有可观的经济效益和社会效益.

关键词: 上位机; 组态设计; 混合逻辑; 顺序控制

Using Full Configuration with Daemon through Upper Computer in Dispatch Center

WENG Yuan

(Guodian Nanjing Automation CO.,LTD, Nanjing 210003, China)

Abstract: Production process control and SCADA systems which based on computer usually handle some complex control process, such as concurrent execution control process, multi-path and multi-condition control process. Based on the current computer and communication technology development level, this paper realizes mixed flow sequential control by provides a method by using full configuration with daemon through upper computer in dispatch center. So that sequential control in lower computer can effectively reduce the shutdown time and operation accidents to achieve much higher reliability, and considerable economic and social benefits.

Keywords: upper computer; configuration design; mixed logic; sequential control

1 引言

顺序控制系统是以计算机为基础的生产过程控制与调度自动化系统. 它可以对各类运行环境、多种运行设备进行监视和控制, 以实现设备控制、数据采集、测量、参数调节、各类信号报警、历史数据存储等各项功能. 随着顺序控制系统的发展, 控制中心针对实时数据的监控已经不能满足于仅仅有单点的遥控, 更期望一次性的顺序下发多点遥控, 也就是所谓的多节点顺序控制, 能够便于中心调度人员操作.

混合流程顺序控制(以下简称顺控)是指通过系统控制界面, 根据生产工艺的要求, 预先规定的操作逻辑和闭锁规则, 自动按规则下发一系列遥控命令, 完成一系列遥控对象的操作, 最终改变系统运行状态的过程, 其控制特点是逐步按条件自动顺序进行, 无需人工干预[1].

2 现状与解决方案

顺控操作最重要的一点就是通过把握好各个遥控节点之间的逻辑顺序关系以及闭锁规则等, 以人机界面的方式提供给用户操作.

现在大部分的顺控系统都是在下位机增加 PLC 控制器以实现顺控的方案[2]. 它所采用的可编程控制器实现了继电器控制所具有的逻辑判断、计时、计数等顺序控制功能[3]. 上位机只发出控制模式号, 该模式的所有逻辑都在下位机的 PLC 实现. 这种方法一方面增加了硬件成本, 且由于下位机地点分散, 若有大规模逻辑变动修改麻烦, 另一方面 PLC 编程具有一定的复杂度, 对于一些现场工程人员的可操作性不强. 这种方法仅适合在设备少、流程路径不多的情况下使用, 否则 PLC 的程序将复杂、冗长, 工艺上每增加一个分

① 收稿时间:2013-10-09 收到修改稿时间:2013-11-18

支(三通), 路径将会增加一倍, 相对应的程序也会增加很多[4]. 所以, 本文着力介绍了一种由上位机监控系统直接组态配置各个遥控对象的控制逻辑顺序, 并通过网络依次下发到下位机, 由下位机顺序执行的设计方法, 现阶段在上位机上实现顺控逻辑操作主要包括两种模式:

2.1 串行顺控操作

目前绝大多数的顺控系统都是串行模式, 根据遥控顺序依次配置各个遥控节点, 在用户下发顺控指令的时候触发其后台顺控程序, 依次读取各个节点并下发相对应的遥控指令[5]. 该模式存在一定局限性, 也就是只有“与”的串行逻辑关系, 只要上一节点遥控失败就一定不能执行下一节点的遥控. 但是这样并不能完全满足用户需求, 在很多的情况下, 顺控操作有多分支并行情况, 也就是既有“与”的逻辑关系, 也有“或”的并行逻辑和“非”的替代逻辑关系, 并且在每个节点遥控执行时都可能会有若干个前提条件, 在此情况下, 该模式不适用; 另外由于该模式逻辑的单一性造成人工干预情况过多, 使得系统智能程度降低[6].

2.2 定制开发的顺控操作

针对第一种模式的缺点, 很多顺控系统的工程项目在顺控操作方面舍弃了组态配置的方法, 而是运用高级开发语言来编写针对单一项目的个性化程序. 其优点是量身定做, 可以满足用户的任何需求. 但是, 该模式也有定制化产品的通病, 开发周期过长, 开发成本过高, 另外由于通用性差, 产品的后期维护, 二次开发等也很浪费人力物力.

针对于此, 本文计从现有技术基础出发, 设计新的全组态多线程多分支的顺控操作方法, 通过界面组态的友好方式配置顺控的各个节点, 并通过在配置节点时的属性参数设置来完成顺控中多分支的“与”、“或”、“非”多重混合逻辑, 由唯一的后台顺控程序来实时读取配置, 并根据配置逻辑依次下发遥控命令, 并可多线程并行下发多个顺控事件. 本文结合了传统的两种顺控操作模式的优点, 适用于任何顺控逻辑, 只需手动进行简单的组态配置, 后台顺控程序唯一且无需更改, 且在整个顺控过程中无需人工干预, 使得顺控系统更加智能化.

3 详细设计与实现

3.1 基本原理

本文采用基于顺控系统平台的组态配置+后台程序+界面触发的设计理念, 其基本工作原理如图 1 所示:

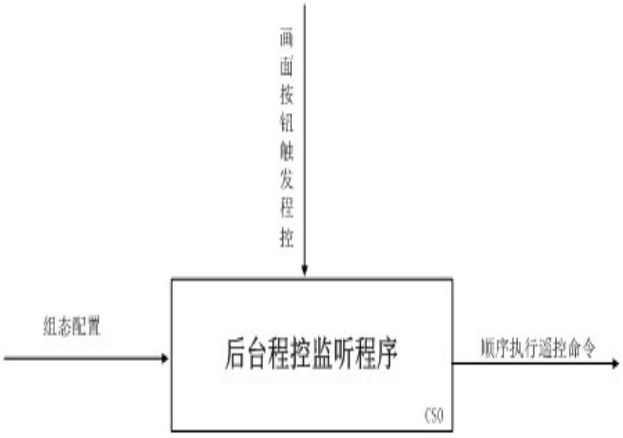

图 1 基本工作原理图

本文能够在使关联到多个遥控开关且有逻辑顺序的遥控操作实现一键运行, 并能够提示用户运行结果. 其操作类型分别如下.

(1) 按照执行逻辑主要可分为 3 类情况.

①顺序执行: 顺控事件执行逻辑为简单串行逻辑, 依次执行顺控的每个步骤, 其工作原理为在执行每个顺控步骤之前都检查上一步执行结果, 执行成功则执行本步骤, 执行失败则停止并退出顺控.

②混合执行: 顺控事件执行逻辑为多重逻辑的混合执行, 在执行顺控步骤时有分支则依次执行所有分支, 若某分支失败则转入下一分支, 若所有分支全部失败则停止并退出顺控. 其工作原理为在执行完每个顺控步骤之后都把执行结果带入一个预先制定好的逻辑表达式中, 该逻辑表达式代表本顺控的实际逻辑关系, 若带入计算结果为“真”则可继续向下执行, 若计算结果为“假”则停止并退出顺控.

③并发执行: 顺控事件执行逻辑为同时下发多个顺控事件, 每个顺控事件相互闭锁, 互不干扰. 其工作原理为通过后台程序下发多线程的方式针对所选择多个顺控事件同时下发执行命令, 并且若多个顺控事件针对同一个顺控对象进行操作, 先操作的顺控事件能够把此对象闭锁, 禁止其他顺控事件操作直至本事件操作结束后解锁.

(2) 按照执行方式主要分为两种方式.

①自动执行: 点击人机界面“自动执行”按钮后所有顺控步骤按照配置逻辑自动执行, 无需人工干预(暂

停和终止除外).

②单步执行: 可人工选择所需执行的顺控步骤, 并点击人机界面“单步执行”按钮后依次执行. 其业务流程图如图 2.

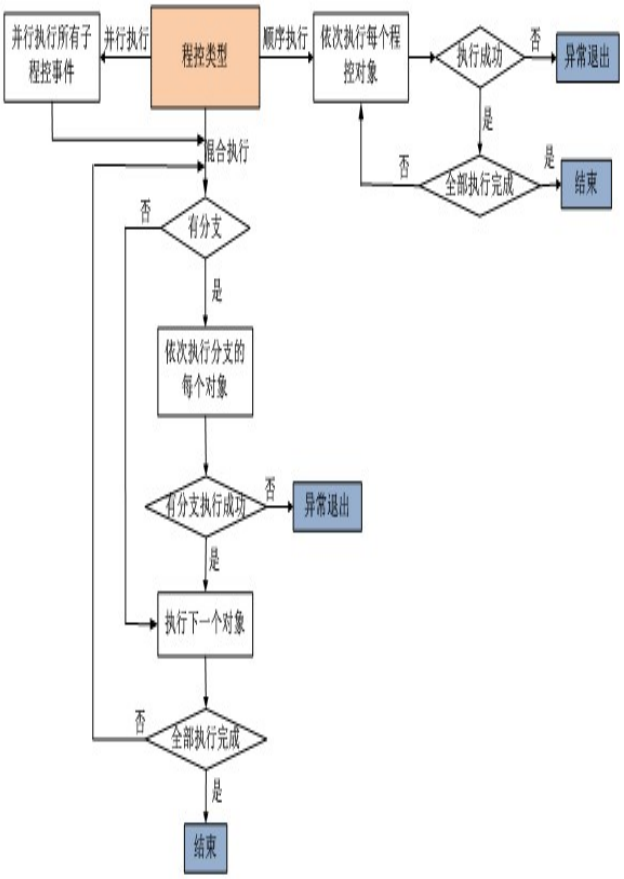

图 2 顺控系统业务流程图

3.2 组态规则

本文基于顺控系统平台, 其组态配置为树形结构, 运用到基于对象、属性的两级组织方式, 分别如下:

3.2.1 对象

本文主要用到的配置对象有如下四个:

• SequenceControl(简称 SC): 顺控事件, 为一系列具体顺控对象的集合.

• SequenceAction(简称 SA): 顺控对象, 为某一个具体的顺控步骤, 关联具体的遥控、遥信对象等.

• DoublePoint(简称 DP): 遥信对象, 用来保存遥控对象返回的状态.

• DoubleCommand(简称 DC): 遥控对象, 用来保存实际需遥控的远方对象, 并关联对应的遥信对象.

3.2.2 属性

本文各个对象包含如下属性(关键属性):

①DoublePoint 对象:

State: 遥信点状态, 例如开、关、故障等等.

②DoubleCommand 对象:

• PointOperationMode: 闭锁条件.

• Command: 下发遥控命令.

• Status: 遥控点执行状态, 例如执行成功、执行失败、执行超时等等.

DoublePointLink: 关联遥信对象.

③SequenceControl 对象:

• Trigger: 触发顺控事件.

• Logic: 逻辑类型, 分为串行、并行、混合型.

• AreaOfResponsibility: 责任区.

• AlarmLink: 关联顺控报警.

④SequenceAction 对象:

• Rank: 顺控对象序号.

• Type: 顺控对象类型, 分为结果、条件、遥控三种.

• ObjectLink: 关联对象, 遥控、遥测等均可.

• ObjectAim: 顺控对象执行目标.

• PreSAExpress: 针对本顺控对象的内部前提条件逻辑表达式.

• CurrentSAExpress: 针对本顺控对象的外部前提条件逻辑表达式.

• Status: 顺控对象执行结果, 例如成功、失败、等待中等待.

3.3 算法实现

根据树形结构的对象配置特点, 本文的后台顺控程序主要思路是从树形图的根节点开始根据递归原理依次展开每个节点, 并且根据每个点的执行情况来判断是否执行下一个点. 树形图每层里的对象无论是 SA 还是 SC, 都按照其 Rank 号依次执行, 其执行方式依赖父对象的 Logic 值、其自身 Logic 和 Status 值以及其子对象的 Status 值.

主要步骤为根据 SC 的 Logic 值以及其子对象的 Status 值来依次执行 SC 的子对象, 若子对象为 SC 则继续查找下一层子对象, 直到子对象为 SA, 在执行 SA 若成功则继续直到该层所有 SA 都执行完毕, 若失败则需要再次判断其父对象的 Logic 属性.

具体流程图如图 3 所示.

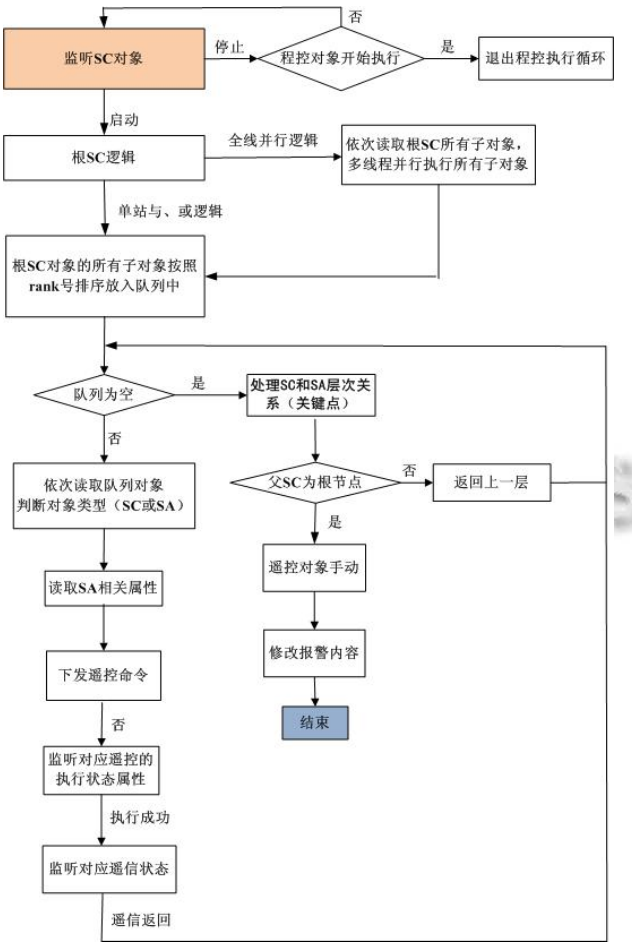

图 3 后台顺控程序流程图

3.4 流程优化

上述方案的优点是配置人员能够根据顺控卡片逻辑灵活配置各种逻辑, 并且顺控步骤任何关键节点出错都能及时跳出顺控. 缺点是本文由于把各种逻辑都显示在配置中(树可能有多层, 且每层都可能既有 SC 也有 SA), 所以对配置人员难度有所加大, 另外因为逻辑复杂, 程序的效率较低. 因此本文还可进行进一步优化, 把递归执行树的节点这个逻辑顺序放入了每个节点的条件逻辑表达式中, 在读取到这个节点时再根据其逻辑表达式结果判断是否执行该节点, 另外设计了一个结果类型的 SA 节点作为顺控步骤的关键节点, 用于作为是否退出顺控的里程碑节点. .

树形图只有根对象为 SC(多线程并发模式例外, 树除了根以外第二层也为SC), 孩子都为SA, 且SA分为三类(条件、遥控、结果), 并且 SA 按照 rank 号排序, 程序读到每个 SA 时都要看其 PreSAExpress 属性所配置的逻辑表达式并确定下一步怎么做.

具体流程图如图 4 所示.

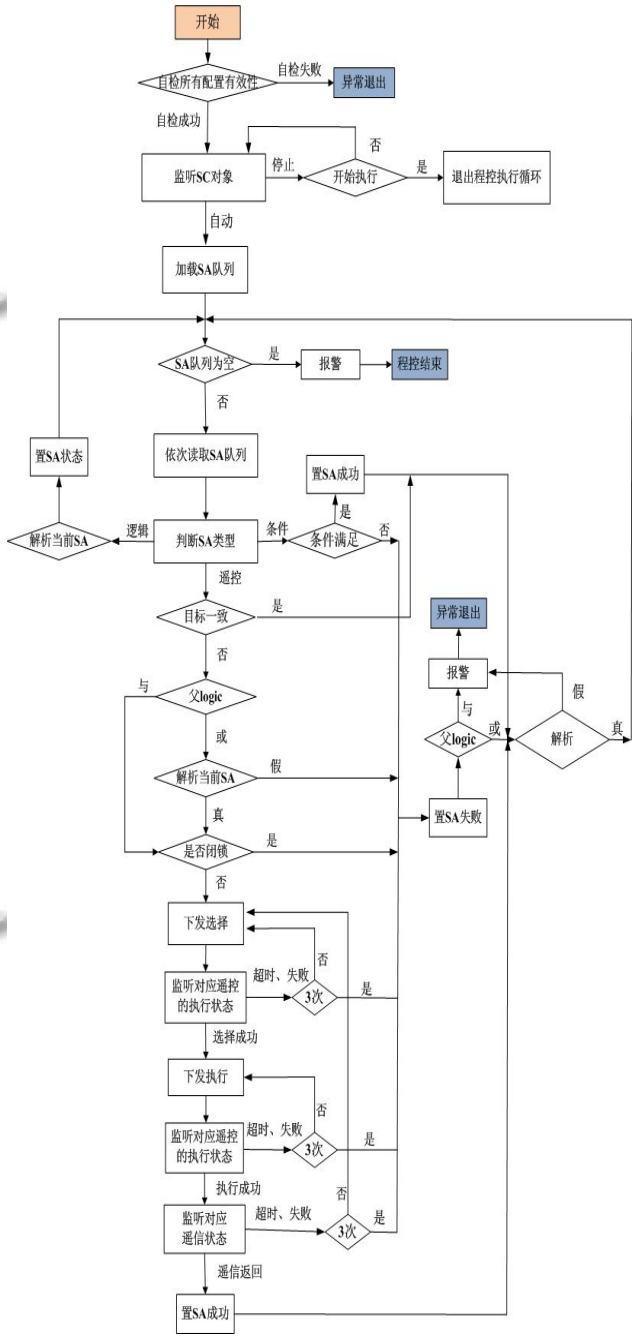

图 4 优化后的后台顺控程序流程图

本文配置简单(只有两层, 第一层为根节点, 第二

层为 SA), 配置人员只需要在 SA 下编写一个逻辑表达式, 此表达式逻辑简单易写, 后台顺控程序只需要判断表达式结果就能决定执行与否, 使得后台程序效率提高, 增加了其稳定性.

3.5 实例

本文设计一个模拟变电所内送电顺控卡片, 顺控卡片如表 1 所示.

表 1　南京地铁主变电所顺控卡片

南京地铁主变电所			
卡片号		1	
项目名称		所内送电	
条件		无	
步骤	站名	开关号	结果
1	元通	1041	合
2	元通	104	合
3	元通	1011	合
4	元通	101	合
5	元通	1021	合
6	元通	102	合

此卡片就代表了一个混合顺控逻辑, 101 和 102 断路器为并行关系, 相互没有闭锁关联. 其电路模拟图如图 5 所示:

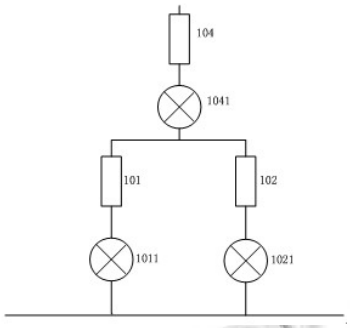

图 5　南京主变电所电力模拟图

根据其逻辑关系, 在配置数据库的组态配置如图 6:

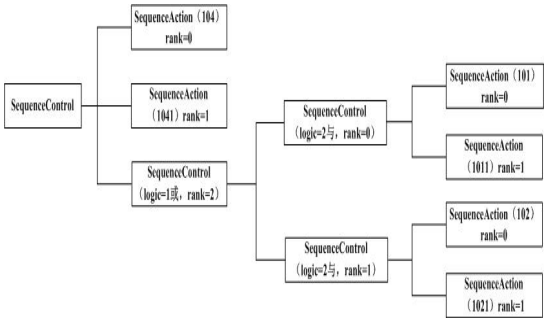

图 6　顺控后台组态配置图

其优化后配置如图 7 所示.

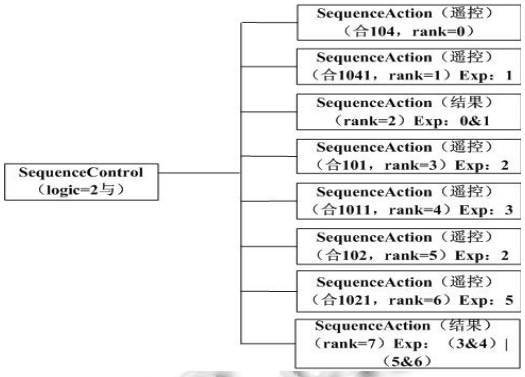

图 7　顺控后台优化后组态配置图

操作界面如图 8 所示:

图 8　顺控后台操作界面

点击自动执行按钮后, 本实例假设 101 断路器遥控失败, 本程控卡片还能够继续执行 102 断路器. 操作有如图 9 所示.

图 9　顺控后台操作结果

3.6 开发工具与源代码

本文实现跨 Windows 平台和 Solaris 平台, 在组态配置方面可基于现在业内任何 SCADA 系统平台, 后台顺控程序和前台操作界面均采用 Eclipse 开发工具. 运用 C++的 Qt 开发包设计开发, Qt 开发包接口丰富易用, 且版本稳定可靠, 并持续更新. Qt 提供了丰富的堆栈容器类功能封装、文件读写及目录操作、线程设计等模块.

本程序关键算法为如何把程控步骤的操作结果带入逻辑表达式中计算, 用到栈和队列的方式来实现后缀表达式的递归算法. 具体源代码如下:

1)程控事件类: CSequenceControl(简称 SC), 当为根节点时表示一张程控卡片, 其子对象(SA, SC 均可)为一系列程控逻辑, 当非根节点时只表示为一系列程控逻辑的父对象, 可理解为虚程控卡片.

具体函数:

CSequenceAction getCurrentAction()

QQueue<QString> getCurrentActionPreExp()

QQueue<QString> getCurrentActionCurrentExp()

CSequenceAction getScanAction()

2)程控对象类: CSequenceAction(简称 SA), 关联实际需遥控对象或遥控条件.

3)主程序: WgSequenceControlApp 主应用程序类, 完成程序的总体管理, 并完成自配置信息初始化, 接收各顺控类传来的命令选择和执行请求, 并将这些请求写入实时库.

具体函数:

bool initializeSequenceControl(CSequenceControl* sequenceControl)

void executeSequenceControl(CSequenceControl* sequenceControl)

void finishSequenceControl(CSequenceControl* sequenceControl)

void terminateSequenceAction(CSequenceControl* sequenceControl)

void scanSequenceControl(CSequenceControl* sequenceControl,int scanMode)

void writeAlarmDescription(int type,WgObId stateStorageObId,QString des)

QQueue<QString>convertSAExpress(QString line)

int SuffixCheck(QQueue<QString> suffixQueue)

4)线程类 CSequenceControlThread: 接收来自程控操作界面来的操作信息并启动程控, 记录日志及报警.

4 结论

本设计方法能适应多个技术领域, 例如电力监控、配网监控、水利监控、油气化工、轨道交通、煤炭矿井等需要进行多节点有逻辑顺序的监视控制业务的工控行业, 尤其适用于电气化铁路和城市轨道交通综合监控系统中的电力监控、环境监控等专业应用, 并由于其灵活的全组态方式, 也可推广到功能需求类似的专业和商业应用, 现在本设计方法已经城市轨道交通综合监控系统等项目中得到了很好的验证.

参考文献

1 吴锦源,梁国坤.顺序控制在变电站倒闸操作中的应用.南方电网技术,2009(3): 46–48.

2 安安.大型顺序控制系统的 PLC 控制方法.第五届全国交通运输领域青年学术会议.2003.226–231.

3 Han ZX. Application of sequence control technique in power plant and discussion on coordinated development of sequence control technique and computer control system. Power System Technology, 2001, 25(10): 63–68.

4 韩朝晖,高晴.基于顺序控制的流程实现方法.工矿自动化,2006,(5): 89–91.

5 佘勇,刘鎏.OPEN 2000 系统在地铁电力监控中的应用.全国电网调度自动化、远动及厂站自动化、电力系统自动化仿真学术,2003: 410–414.

6 王秋娜,谢承.基于组态软件电力监控系统软件设计.信息系统工程,2012,(4):18－22.