(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 109933011 A
(43)申请公布日 2019.06.25

(21)申请号 201711345135.1

(22)申请日 2017.12.15

(71)申请人 中国科学院沈阳自动化研究所
地址 110016 辽宁省沈阳市东陵区南塔街114号

(72)发明人 刘文成 王挺 王戬 于海斌 曾鹏

(74)专利代理机构 沈阳科苑专利商标代理有限公司 21002
代理人 王倩

(51)Int.Cl.
*G05B 19/418*(2006.01)

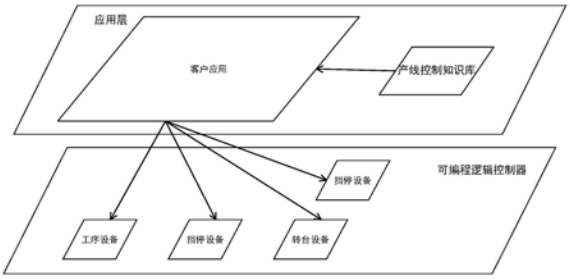
权利要求书2页 说明书8页 附图6页

(54)发明名称

一种基于面向对象思想的产线控制知识库建立方法

(57)摘要

本发明涉及一种基于面向对象思想的产线控制知识库建立方法，将面向对象思想与IEC61499编程标准相结合，从而建立能够实现柔性生产的产线控制知识库。包括以下步骤：功能块间事件驱动接口定义，生产订单获取功能块的建立，挡停控制功能块的建立、转台控制功能块的建立以及工序控制功能块的建立。本发明专利能够将底层产线控制抽象，建立一个不断完善的通用知识库，从而最终达到应用层以拖拽、配置的方式完成产线控制，减少生产人员的工作量，提高工作效率，增加了生产灵活性，能够实现离散制造行业的生产线快速重构。




CN 109933011 A

1.一种基于面向对象思想的产线控制知识库建立方法,其特征在于,通过建立一种或多种功能块构成知识库,所述功能块的建立如下:

根据事件的逻辑定义功能块的输入接口和输出接口,用于实现功能块之间、功能块与产线间的关联;其中,将事件的输入、变量值作为功能块的输入接口,用于输入工艺信息;将事件的输出、状态值作为功能块的输出接口,用于输出设备命令和工作状态。

2.根据权利要求1所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于所述功能块的建立还包括以下步骤:

建立功能块与产线系统的通信;

定义与产线系统通信的格式;

根据产线和工艺实现功能块的内部逻辑,通过事件触发。

3.根据权利要求1所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述功能块按类型分为生产订单获取功能块、挡停控制功能块、转台控制功能块、工序控制功能块。

4.根据权利要求3所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述生产订单获取功能块内部逻辑的建立包括以下步骤:

步骤1)建立生产线与ERP系统的socket通信协议连接;

步骤2)建立通信格式;

步骤3)生产订单的上线控制:当订单个数小于N时,生产订单获取功能块从ERP系统请求获得订单号,并投入上线;N为可配置的订单个数;

步骤4)订单上线控制的封装:首先触发输出事件reqNextState,接着连续接收输入事件recvNextState,如果后续设备状态nextState为空闲状态,则进行订单上线。

5.根据权利要求3所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述挡停控制功能块的建立包括以下步骤:

步骤1)接收订单号:触发挡停控制功能块接收订单号recvID事件,当传感器接收到托盘到达信号时,输出sendArrive事件;

步骤2)接收到达信号:触发挡停控制功能块接收托盘到达下一个设备信号recvArrive事件,重置内部变量到达标识arrive=1;

步骤3)挡停放行:触发挡停控制功能块放行go事件,输出goComplete事件,置当前挡停状态localstate为空闲;

步骤4)接收加工工位加工完成:触发挡停控制功能块recvReqDown事件,输出可以传送至挡停sendCmd事件;

步骤5)接收状态请求:触发挡停控制功能块接收状态请求recvReq事件,输出outputState事件返回挡停状态。

6.根据权利要求3所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述转台控制功能块的建立如下:

步骤1)处理转向模块的建立:根据产品的订单号以及当前产线的工艺路线来判断转台的转向;

步骤2:传感器服务模块的建立:如果监听到托盘到位信号,输出订单号给处理转向服务块,内部变量localstate置为1忙碌,并输出sendArrive1或sendArrive2事件;如果监听

到托盘离开的信号,则置内部变量localstate为0空闲,置转台挡停为挡停状态;

步骤3:处理接收数据模块的建立:触发recvArrive1或recvArrive2事件,重置内部变量arrive1或arrive2为1,表示1输出口托盘已经到达下一个设备,2输出口的托盘已经到达下一个设备;触发recvID1或recvID2事件,重置内部变量订单号ID1或订单号ID2,表示1输入口即将获得订单号为ID1的产品,2输入口即将获得订单号为ID2的产品;

步骤4:转台旋转模块的建立:根据输入口以及输出口的转向设定旋转角度;

步骤5:处理接收状态请求模块的建立:触发处理状态请求事件recvReq1或recvReq2;根据转台的当前角度和状态判断空闲或忙碌并输出;

步骤6:转台放行模块的建立:获得处理转向模块的放行输入口,一旦放行成功输出goComplete事件。

7.根据权利要求6所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述处理转向模块的建立如下:

步骤1.1:建立转台控制功能块与生产车间MES系统的socket通信;

步骤1.2:通信格式的建立;

步骤1.3:完成接收信息的解析,获得输出口ID。

8.根据权利要求3所述的一种基于面向对象思想的产线控制知识库建立方法,其特征在于,所述工序控制功能块的建立包括以下步骤:

步骤1:建立与MES系统的socket通信;

步骤2:定义与MES系统的通信格式;

步骤3:建立与加工机器人的socket通信;

步骤4:建立工序加工过程:当产品到达加工工位后不需要加工,则输出noAssembel事件;如果需要则继续通过socket通信协议获得机器人指令,并发送给机器人进行加工,当获得加工完成信号时,则输出reqDown事件,请求传输回挡停;当下一个挡停设备同意时,输出complete事件,完成加工过程。

# 一种基于面向对象思想的产线控制知识库建立方法

## 技术领域

[0001] 本发明涉及了一种基于面向对象思想的产线控制知识库建立方案，属于智能制造控制领域。

## 背景技术

[0002] 随着社会的发展，生产水平不断提高，技术更新越来越快，生产模式逐渐从过去的大批量单一品种生产转为变批量多品种的柔性生产，传统产线面临着严峻挑战，如何适应这种生产模式变换，降低切换产线、柔性生产的成本是一个亟待解决的问题。

[0003] 传统产线中，产线的搭建需要多个可编程逻辑控制器，可编程逻辑控制器起着至关重要的作用，可编程控制器有着运行稳定，响应速度快等特点。在生产过程中，可编程控制器负责控制生产逻辑，处理订单信息等工作。但是可编程控制器中的代码存在众多冗余。并且切换新工艺，重构生产线时，需要重新编写可编程逻辑控制器中的代码，浪费大量时间，降低了生产效率。并且，可编程控制器中的代码需要专业的PLC编程技术人员进行编写，普通的生产者无法参与到产线的定制中，大大提高了维护成本，降低生产利润。

[0004] 本发明提出了一种基于面向对象思想的产线控制知识库建立方案。首先对产线的所有设备的处理过程进行抽象，抽象出四种逻辑，分别是生产订单获取逻辑、挡停逻辑、转台逻辑以及工序逻辑。接着对这四种逻辑进行封装，求解最大集合，构成一个通用产线控制知识库，保证第三方能够高效利用，减少重构生产线时可编程逻辑控制器的编程，并且使得编程方式更加简单，这样普通的生产线管理人员可以加入到产线的重构中来，不必再让技术人员花费大量时间来调整生产线加工工艺，提高了生产效率，降低了生产成本。弱化了可编程控制器的编程语言，将可编程控制器的开发语言作为一个底层知识库，通过上层高级语言的编程来完成可编程逻辑控制器的编程。本发明从多方面出发，实现了高效率重构产线的要求，并且提高了生产效率，降低了生产成本。

## 发明内容

[0005] 本发明针对传统生产线的可编程逻辑控制器编程的局限性，难以应对目前离散加工产品的变批量多品种加工特点，无法对生产线进行快速重构调整，耗费人力成本较高，影响加工进度，提出了一种基于面向对象思想的产线控制知识库建立方案。将生产线设备的加工特征进行抽象，利用IEC61499编程标准的特点，采用事件驱动的模型进行对象抽象封装，对加工过程进行解耦合，进行独立封装，并提供调用接口。在切换产线过程中，只需要在应用层利用高级语言简单的拖拽式编程即可进行生产线逻辑的重构，提高了生产效率、降低了生产线的维护成本。

[0006] 本发明采用的技术方案如下：一种基于面向对象思想的产线控制知识库建立方法，通过建立一种或多种功能块构成知识库，所述功能块的建立如下：

[0007] 根据事件的逻辑定义功能块的输入接口和输出接口，用于实现功能块之间、功能块与产线间的关联；其中，将事件的输入、变量值作为功能块的输入接口，用于输入工艺信

息;将事件的输出、状态值作为功能块的输出接口,用于输出设备命令和工作状态。

[0008] 所述功能块的建立还包括以下步骤:

[0009] 建立功能块与产线系统的通信;

[0010] 定义与产线系统通信的格式;

[0011] 根据产线和工艺实现功能块的内部逻辑,通过事件触发。

[0012] 所述功能块按类型分为生产订单获取功能块、挡停控制功能块、转台控制功能块、工序控制功能块。

[0013] 所述生产订单获取功能块内部逻辑的建立包括以下步骤:

[0014] 步骤1)建立生产线与ERP系统的socket通信协议连接;

[0015] 步骤2)建立通信格式;

[0016] 步骤3)生产订单的上线控制:当订单个数小于N时,生产订单获取功能块从ERP系统请求获得订单号,并投入上线;N为可配置的订单个数;

[0017] 步骤4)订单上线控制的封装:首先触发输出事件reqNextState,接着连续接收输入事件recvNextState,如果后续设备状态nextState为空闲状态,则进行订单上线。

[0018] 所述挡停控制功能块的建立包括以下步骤:

[0019] 步骤1)接收订单号:触发挡停控制功能块接收订单号recvID事件,当传感器接收到托盘到达信号时,输出sendArrive事件;

[0020] 步骤2)接收到达信号:触发挡停控制功能块接收托盘到达下一个设备信号recvArrive事件,重置内部变量到达标识arrive=1;

[0021] 步骤3)挡停放行:触发挡停控制功能块放行go事件,输出goComplete事件,置当前挡停状态localstate为空闲;

[0022] 步骤4)接收加工工位加工完成:触发挡停控制功能块recvReqDown事件,输出可以传送至挡停sendCmd事件;

[0023] 步骤5)接收状态请求:触发挡停控制功能块接收状态请求recvReq事件,输出outputState事件返回挡停状态。

[0024] 所述转台控制功能块的建立如下:

[0025] 步骤1)处理转向模块的建立:根据产品的订单号以及当前产线的工艺路线来判断转台的转向;

[0026] 步骤2:传感器服务模块的建立:如果监听到托盘到位信号,输出订单号给处理转向服务块,内部变量localstate置为1忙碌,并输出sendArrive1或sendArrive2事件;如果监听到托盘离开的信号,则置内部变量localstate为0空闲,置转台挡停为挡停状态;

[0027] 步骤3:处理接收数据模块的建立:触发recvArrive1或recvArrive2事件,重置内部变量arrive1或arrive2为1,表示1输出口托盘已经到达下一个设备,2输出口的托盘已经到达下一个设备;触发recvID1或recvID2事件,重置内部变量订单号ID1或订单号ID2,表示1输入口即将获得订单号为ID1的产品,2输入口即将获得订单号为ID2的产品;

[0028] 步骤4:转台旋转模块的建立:根据输入口以及输出口的转向设定旋转角度;

[0029] 步骤5:处理接收状态请求模块的建立:触发处理状态请求事件recvReq1或recvReq2;根据转台的当前角度和状态判断空闲或忙碌并输出;

[0030] 步骤6:转台放行模块的建立:获得处理转向模块的放行输入口,一旦放行成功输

出goComplete事件。

[0031] 所述处理转向模块的建立如下：

[0032] 步骤1.1:建立转台控制功能块与生产车间MES系统的socket通信；

[0033] 步骤1.2:通信格式的建立；

[0034] 步骤1.3:完成接收信息的解析,获得输出口ID。

[0035] 所述工序控制功能块的建立包括以下步骤：

[0036] 步骤1:建立与MES系统的socket通信；

[0037] 步骤2:定义与MES系统的通信格式；

[0038] 步骤3:建立与加工机器人的socket通信；

[0039] 步骤4:建立工序加工过程:当产品到达加工工位后不需要加工,则输出noAssembel事件;如果需要则继续通过socket通信协议获得机器人指令,并发送给机器人进行加工,当获得加工完成信号时,则输出reqDown事件,请求传输回挡停;当下一个挡停设备同意时,输出complete事件,完成加工过程。

[0040] 本发明的优点是：

[0041] 1)生产线的快速重构。重构包括,更改加工工位位置、新增加工工位。过程中只需要短暂停线,在应用层直接调用产线控制知识库,快速构成完整的程序后统一编译下载到可编程逻辑控制器中即可完成重构。

[0042] 2)为多品种变批量生产提供底层支撑,能够快速响应生产线的更改的同时,能够保证生产线的可编程逻辑控制器识别同一生产线的不同产品,保证生产线上的混线生产。

[0043] 3)生产线的机器人更新换代或者提高产能,均可以快速对生产线做出响应,提高了生产工作效率。

## 附图说明

[0044] 图1是本发明生产线控制的整体框架示意图。

[0045] 图2是本发明获取订单功能块接口定义示意图。

[0046] 图3是本发明挡停功能块接口定义示意图。

[0047] 图4是本发明转台功能块接口定义示意图。

[0048] 图5是本发明工序控制功能块接口定义示意图。

[0049] 图6是本发明订单投入上线产能控制算法流程图。

[0050] 图7是本发明订单上线算法流程图。

[0051] 图8是本发明转台控制功能块中模块结构示意图。

[0052] 图9是本发明工序控制功能块加工过程算法示意图。

## 具体实施方式

[0053] 为使本发明的目的、技术方案及优点更加清楚、明确,下面参照附图进行详细说明。

[0054] 本发明能够将底层产线控制逻辑抽象,建立一个不断完善的通用知识库,从而最终达到应用层以拖拽、配置的方式完成产线控制编程,减少生产人员的工作量,提高工作效率,增加了生产灵活性,能够实现离散制造行业的生产线快速重构。

[0055] 本方案采用如下步骤来完成产线控制知识库的建立：

[0056] 1)功能块间事件驱动接口定义。

[0057] 2)生产订单获取功能块的建立。

[0058] 3)挡停控制功能块的建立。

[0059] 4)转台控制功能块的建立。

[0060] 5)工序控制功能块的建立。

[0061] 其中,功能块间事件驱动接口定义的具体步骤是：

[0062] 1)生产订单获取功能块的接口定义。生产订单获取功能块与挡停功能块相关联。

[0063] 2)挡停控制功能块的接口定义。挡停控制功能块可与挡停控制功能块、工序控制功能块、转台控制功能块、以及生产订单获取功能块相关联。

[0064] 3)转台控制功能块的接口定义。转台控制功能块可与挡停控制功能块相关联。

[0065] 4)工序控制功能块的接口定义。工序控制功能块可与挡停控制功能块相关联。

[0066] 其中,生产订单获取功能块建立的具体方法是：

[0067] 1)封装socket连接,保证生产线与ERP系统能够进行通信。

[0068] 2)与ERP系统通信格式建立。

[0069] 3)生产订单上线控制算法的建立。

[0070] 4)订单上线算法的封装。

[0071] 其中,挡停控制功能块建立的具体步骤是：

[0072] 1)挡停获得订单号处理算法。

[0073] 2)挡停接收到达信号处理算法。

[0074] 3)挡停放行算法。

[0075] 4)挡停接收加工工位加工完成信号算法。

[0076] 5)挡停接收状态请求算法。

[0077] 其中,转台控制功能块建立的具体步骤是：

[0078] 1)处理转向模块的建立。

[0079] 2)传感器服务模块的建立。

[0080] 3)处理接收数据模块的建立。

[0081] 4)转台旋转模块的建立。

[0082] 5)处理接收状态请求模块的建立。

[0083] 6)转台放行模块的建立。

[0084] 其中,工序控制功能块建立的具体步骤是：

[0085] 1)建立与MES系统的socket通信。

[0086] 2)与MES系统通信格式的定义。

[0087] 3)建立与加工机器人的socket通信。

[0088] 4)工序加工过程算法的建立。

[0089] 参见附图1,是本发明的整体架构图,产线控制知识库包括获取订单功能块、挡停控制功能块、转台控制功能块以及工序控制功能块。产线控制知识库是可编程逻辑控制器中逻辑代码的封装,基于面向对象思想以及IEC61499编程规范,实现产线逻辑控制的高级应用,为应用层语义化编程提供研究基础。减少生产人员的工作量,提高工作效率,增加了

生产灵活性，能够实现离散生产线的快速重构。不同种类的功能块之间存在基于事件驱动的接口，一个生产线逻辑控制程序需要多种类的功能块共同构成。

[0090] 下面，针对产线控制知识库中不同功能块的建立以及功能块之间的接口定义过程作详细说明。

[0091] 首先，四种功能块间事件驱动接口的定义。生产线控制逻辑的建立需要多类控制功能块的协调完成。功能块间的连接采用基于事件驱动的信息传递。功能块间事件驱动接口定义的步骤如下。

[0092] 步骤1：生产订单获取功能块的接口定义。生产订单获取功能块与挡停功能块相关联。如图2所示，为生产订单获取功能块接口示意图。输入事件有“接收下一个挡停状态recvNextState”，“产线开始sysStart”，“获得工艺ID getProcessId”，输出事件有“发送订单号sendId”，“请求下一个挡停状态reqNextState”，“订单初始化生产init”。

[0093] 步骤2：挡停控制功能块的接口定义。挡停控制功能块可与挡停控制功能块、工序控制功能块、转台控制功能块以及生产订单获取功能块相关联。如图3所示，为挡停控制功能块接口示意图。输入事件有“接收订单号recvID”，“接收状态请求recvReq”，“接收放行指令go”，“接收到达指令recvArrive”，“接收完成传送到加工工序指令completeRise”，“接收完成指令recvReqDown”，“接收下一个挡停状态recvNextState”。输出事件有“托盘到位指令palletIn”，“请求下一个挡停状态指令reqNextState”，“放行完成指令goComplete”，“输出状态指令outputState”，“发送到达指令sendArrive”，“发送加工指令sendCmd”。

[0094] 步骤3：转台控制功能块的接口定义。转台控制功能块可与挡停控制功能块相关联。如图4所示，为转台控制功能块接口示意图。假定生产线中的转台设备存在两个进入方向，两个放行方向。输入事件有两个方向的“接收到达指令recvArrive”，两个方向的“接收下一个挡停状态指令recvNextState”，两个方向的“接收订单号recvID”，两个方向的“接收上一个状态请求recvReq”。输出事件有两个方向的“发送到达指令sendArrive”，两个方向的“放行完成指令goComplete”，两个方向的“请求下一个设备状态指令reqNextState”，两个方向的“输出当前状态指令outputState”。

[0095] 步骤4：工序控制功能块的接口定义。工序控制功能块可与挡停控制功能块相关联。如图5所示，为工序控制功能块接口示意图。输入事件有“托盘到位指令palletIn”，“接收可以传送至挡停指令recvDown”，输出事件有“完成指令complete”，“不装配指令noAssemble”，“请求下降指令reqDown”，“到达工位通知指令completeRise”。

[0096] 图2-图5中的其它接口表示输入变量值、输出状态值。

[0097] 接着，生产订单获取功能块的建立。与ERP系统交互获得生产订单是生产线的驱动，是生产线的起始部分，负责统筹管理整个产线的加工订单，从ERP系统数据库中不断获取订单，投入生产。获得生产订单功能块建立步骤如下：

[0098] 步骤1：建立生产线与ERP系统的socket通信协议连接，通过socket通信协议与ERP系统建立信息交互。

[0099] 步骤2：通信格式的建立。生产线发送信息格式为[车间ID,产线ID,工艺ID\n]，其中车间ID为自定义车间编号，产线ID为生产线自定义编号，工艺ID为当前正在加工的产品所属工艺的编号。ERP系统通过socket通信协议获得生产线发送的信息进行解析后返回的信息格式为[工艺ID,订单号]。

[0100] 步骤3:生产订单上线控制。如图6所示,获得订单号,并投入上线算法示意图。当生产线启动,首先判断在线的订单个数,当大于等于N时,停止获得订单。当订单个数小于N时,获取生产订单功能块需要从ERP系统请求获得订单号,并投入上线。N为可配置的订单个数,客户根据生产线的产能自行设置订单个数。

[0101] 步骤4:订单上线算法封装。如图7所示。首先触发输出事件reqNextState,接着不断接收输入事件recvNextState,如果后续设备状态nextState为空闲状态,则进行订单上线。

[0102] 然后,挡停控制功能块的建立。挡停控制功能块即是产线控制中的挡停设备的逻辑封装,是生产线物流系统必不可少的设备之一。需要处理托盘到位检测、挡停、放行等事件。挡停控制功能块可与挡停控制功能块、工序控制功能块、转台控制功能块以及订单获取功能块相关联。

[0103] 挡停控制功能块的建立步骤如下:

[0104] 步骤1:接收订单号处理算法。触发挡停功能块接收订单号recvID事件,即是前一个设备已经放行,此时挡停功能块需要重置内部变量订单号ID＝IDUnfinished。并且开启挡停传感器监听,不断监听传感器的输出信号值,判断托盘是否到达当前挡停。一旦监听托盘到位信号,并且判断reqDown＝0,则输出palletIn事件,并输出sendArrive事件,并将localsState置为1(忙碌)。当reqDown＝1时,仅将localstate置为1(忙碌),直接请求挡停放行。

[0105] 步骤2:接收到达信号处理算法。触发挡停功能块接收托盘到达下一个设备信号recvArrive事件。此时挡停功能块需要重置内部变量到达标识arrive＝1。

[0106] 步骤3:挡停放行算法。触发挡停功能块放行go事件。即挡停获得放行指令,挡停放行条件为(arrive＝1&&ID!＝0&&localstate＝忙碌&&nextstate＝空闲),一旦放行之后,ID＝0,arrive＝0,如果reqDown＝1则置为0。并且开启挡停传感器监听,不断监听传感器的输出信号值,判断托盘是否离开当前挡停。当监听到托盘离开信号,判断arrive是否为1,如果arrive为1,则将localstate＝空闲,arrive＝1,ID＝0,reqDown＝0。否则挡停状态localstate置为0空闲。

[0107] 步骤4:接收加工工位加工完成信号算法。触发挡停功能块recvReqDown事件,即加工工位加工完成信号。此时置reqDwon为1。不断判断内部变量ID是否为0,如果ID＝0则输出可以传送至挡停sendCmd事件。

[0108] 步骤5:接收状态请求算法。触发挡停功能块接收状态请求recvReq事件。即前一个设备请求当前挡停的状态,获得是空闲还是占用状态。当内部变量reqDown＝0&&localState＝空闲时,输出outputState事件返回挡停空闲。当条件不满足时,输出outputState事件,返回挡停占用。

[0109] 接着,转台控制功能块的建立。转台是生产线中的重要物流设备。保证产品在生产线按顺序进行加工生产。在原有的可编程逻辑控制器中,转台控制逻辑很复杂。需要耗费大量时间进行冗余编写。本发明针对这一问题,对转台逻辑进行封装,为应用层的语义编程提供基础。转台功能块的内部模块如图8所示。本产线控制知识库中仅能处理存在两个进入方向,两个放行方向的转台。

[0110] 转台控制功能块的建立步骤如下:

[0111] 步骤1:处理转向模块的建立。转向模块根据产品的订单号以及当前产线的工艺路线来判断转台的转向,使得产品能够有序地在生产线进行加工生产。

[0112] 步骤1.1:建立转台功能块与生产车间MES系统的socket通信,通过socket通信协议与MES系统进行信息交互。

[0113] 步骤1.2:通信格式的建立。转台功能块的发送信息格式为[转台ID,订单号ID,生产工艺ID]。其中转台ID为转台的唯一编号,订单号为订单编号,生产工艺号为当前订单的生产工艺编号。转台功能块接收信息的格式为[订单号ID,输出口ID],其中输出口ID即为转台需要转向放行的出口编号。

[0114] 步骤1.3:信息的解析算法。转向功能块能够完成步骤1.2中对接收信息的解析,获得输出口ID。

[0115] 步骤2:传感器服务模块的建立。不断监听传感器输入值,如果监听到托盘到位信号,与转向模块关联,输出订单号给处理转向服务块,localstate置为1忙碌,并输出sendArrive1或sendArrive2事件。如果监听到托盘离开的信号,内部变量localstate置为0空闲,转台挡停置为挡停状态。

[0116] 步骤3:处理接收数据模块的建立。触发recvArrive1或recvArrive2事件,转台功能块需要分别重置内部变量arrive1或arrive2为1,表示1输出口托盘已经到达下一个设备,2输出口的托盘已经到达下一个设备。触发recvID1和recvID2事件,转台功能块分贝重置内部变量ID1和ID2,表示1输入口即将获得订单号为ID1的产品,2输入口即将获得订单号为ID2的产品。

[0117] 步骤4:转台旋转模块的建立。本产线控制知识库仅能够处理两个输入口,两个输出口。所以转台中输入口以及输出口的的角度固定。输入口1转台处于0度状态。输入口2需要转台处于-90度状态。输出口1需要转台处于0度状态,输出口2需要转台处于90度状态,其中顺时针方向为正,逆时针角度为负,转台的默认角度为0度状态,转台旋转模块根据其他模块的指令,与旋转设备进行通信,进行一定角度的旋转。

[0118] 步骤5:处理接收状态请求模块的建立。触发处理状态请求事件recvReq1和recvReq2。如果触发recvReq1的事件,则判断如果当前角度angle＝＝0&&当前状态localState为空闲,则输出空闲。否则输出忙碌,并且如果angle!＝0&&localstate为空闲则与转台旋转模块通信,使转台旋转至0度。如果触发recvReq2的事件,则判断如果当前角度angle＝＝-90&&当前状态localState为空闲,则输出空闲。否则输出忙碌,并且如果angle!＝-90&&localstate为空闲则与转台旋转模块通信,使转台旋转至-90度。

[0119] 步骤6:转台放行模块的建立。获得处理转向模块的放行输出口,如果获得输出口1,则判断当前转台角度是否为0度,并且nextState＝空闲&&arrive＝1,如果angle!＝0&&nextState＝＝空闲&&arrive＝1,则与转台旋转模块通信,让转台旋转至0度;如果获得输出口2,则判断当前转台角度是否为90度,并且nextState＝空闲&&arrive＝1,如果angle!＝90&&nextState＝＝空闲&&arrive＝1,则与转台旋转模块通信,让转台旋转至90度。一旦放行成功输出goComplete事件,置内部变量arrive＝0,nextstate＝忙碌。

[0120] 最后,工序控制功能块的建立。工序控制功能块是生产线的加工控制。加工机器人以及装配手是加工的重要设备,并且种类繁多,原有的可编程逻辑控制器代码复杂。本发明对工序控制逻辑进行封装,解耦合。工序控制功能块建立的具体步骤如下:

[0121] 步骤1:建立与MES系统的socket通信。通过与MES系统的信息交互,将订单号包装成当前工位加工机器人可以识别的信息,发送给加工机器人,从而实现了产线控制与生产加工的解耦合。

[0122] 步骤2:与MES系统通信格式的定义。工序控制功能块发送消息的格式定义为[工位ID,工艺ID,订单ID],其中工位ID为工位的唯一编号,工艺ID为产品工艺编号,订单ID为订单的唯一编号。工序控制功能块的接收消息为机器人可以识别的自定义格式字符串。

[0123] 步骤3:建立与加工机器人的socket通信。工序控制功能块获得MES系统发送的自定义格式字符串,通过socket通信与加工机器人进行交互,启动加工机器人进行加工。

[0124] 步骤4:工序加工过程算法的建立。如图9所示,为工序控制功能块加工过程算法流程图。当产品到达加工工位后,首选需要从ERP系统判断当前产品是否需要在当前工位进行加工,如果不需要加工,则输出noAssembel事件;如果需要则继续通过socket通信协议获得机器人指令,并发送给机器人进行加工。当获得加工完成信号时,则输出reqDown事件,请求传输回挡停。当关联的挡停设备同意时,输出complete事件,完成加工过程。

[0125] 以上内容阐述了一种基于面向对象思想的产线控制知识库的设计及实现方法。本发明针对目前产线在变批量多品种生产中的不足,且生产过程中无法快速调整生产线结构,花费大量人力物力成本等问题,提出了产线控制知识库建立方法,减少重构生产线时可编程逻辑控制器的编程,并且使得编程方式更加简单,减少了人力成本。同时也为之后的生产线语义化编程的研究提供了底层支撑。

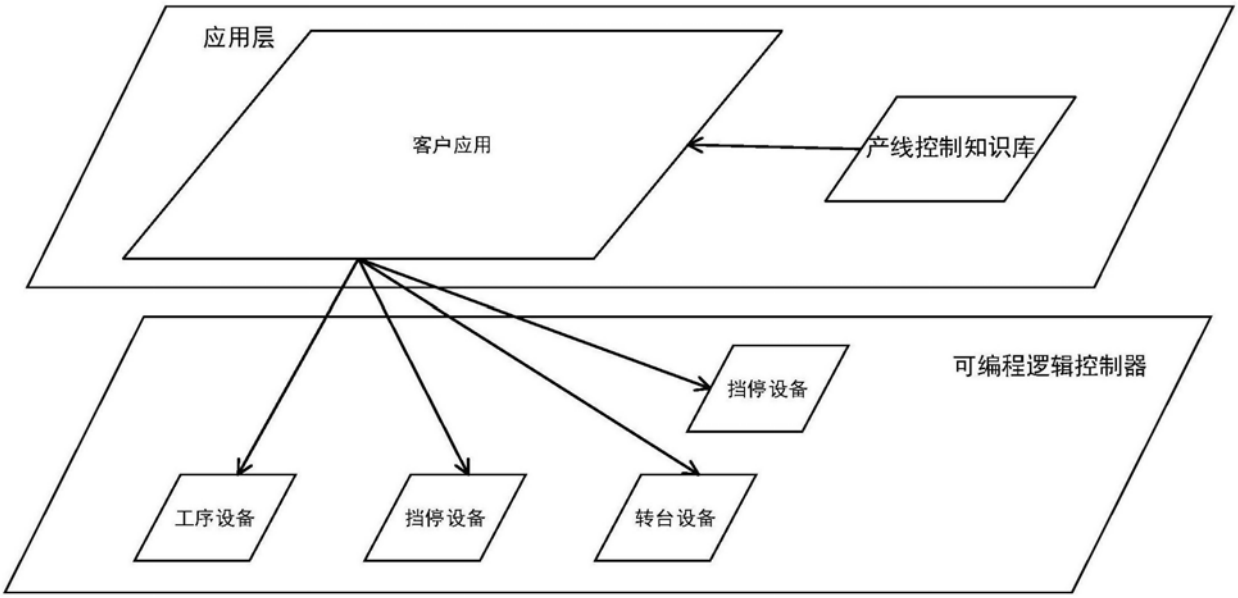

应用层
客户应用
产线控制知识库
可编程逻辑控制器
挡停设备
工序设备
挡停设备
转台设备

图1


获取订单功能块
sysStart
getProcessId
recvNextState
processId
nextState
sendId
reqNextState
init
ID

图2

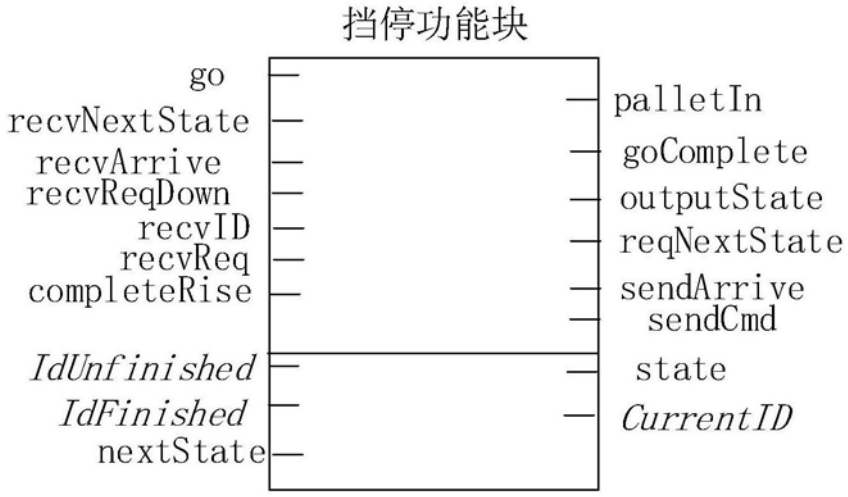

挡停功能块
go
recvNextState
recvArrive
recvReqDown
recvID
recvReq
completeRise
IdUnfinished
IdFinished
nextState
palletIn
goComplete
outputState
reqNextState
sendArrive
sendCmd
state
CurrentID

图3

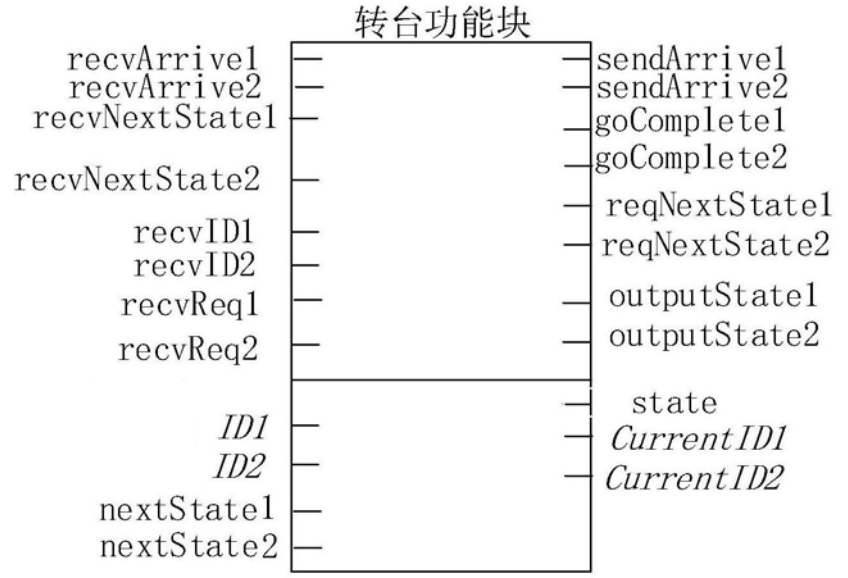

转台功能块
recvArrive1
recvArrive2
recvNextState1
recvNextState2
recvID1
recvID2
recvReq1
recvReq2
ID1
ID2
nextState1
nextState2
sendArrive1
sendArrive2
goComplete1
goComplete2
reqNextState1
reqNextState2
outputState1
outputState2
state
CurrentID1
CurrentID2

图4

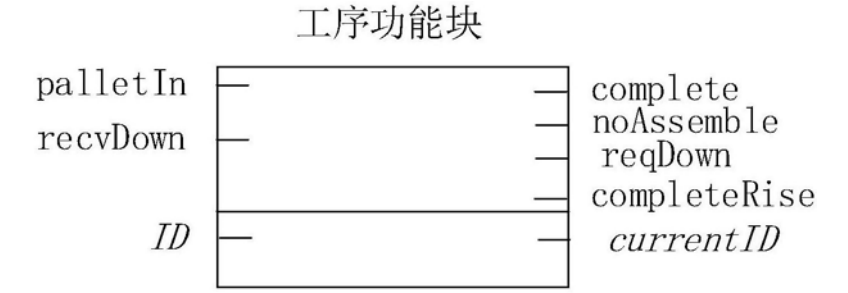

工序功能块
palletIn
recvDown
ID
complete
noAssemble
reqDown
completeRise
currentID

图5

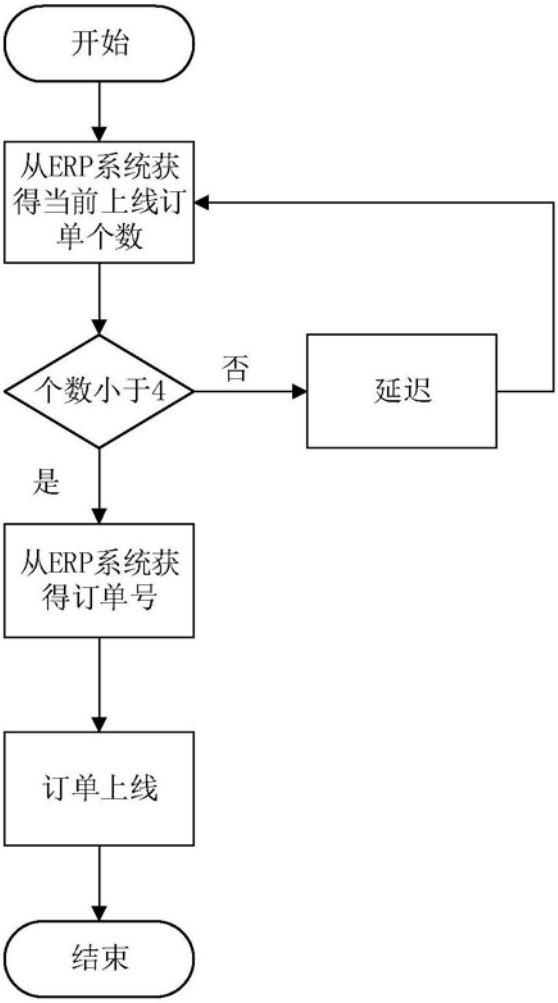



图6

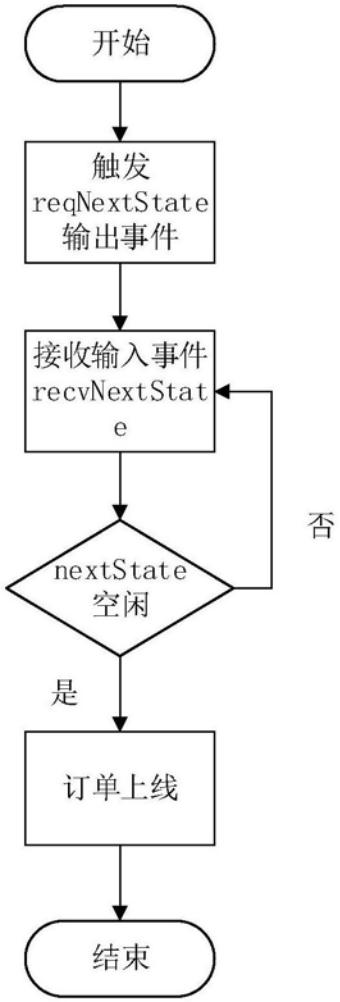

开始
触发
reqNextState
输出事件
接收输入事件
recvNextStat
e
nextState
空闲
否
是
订单上线
结束

图7

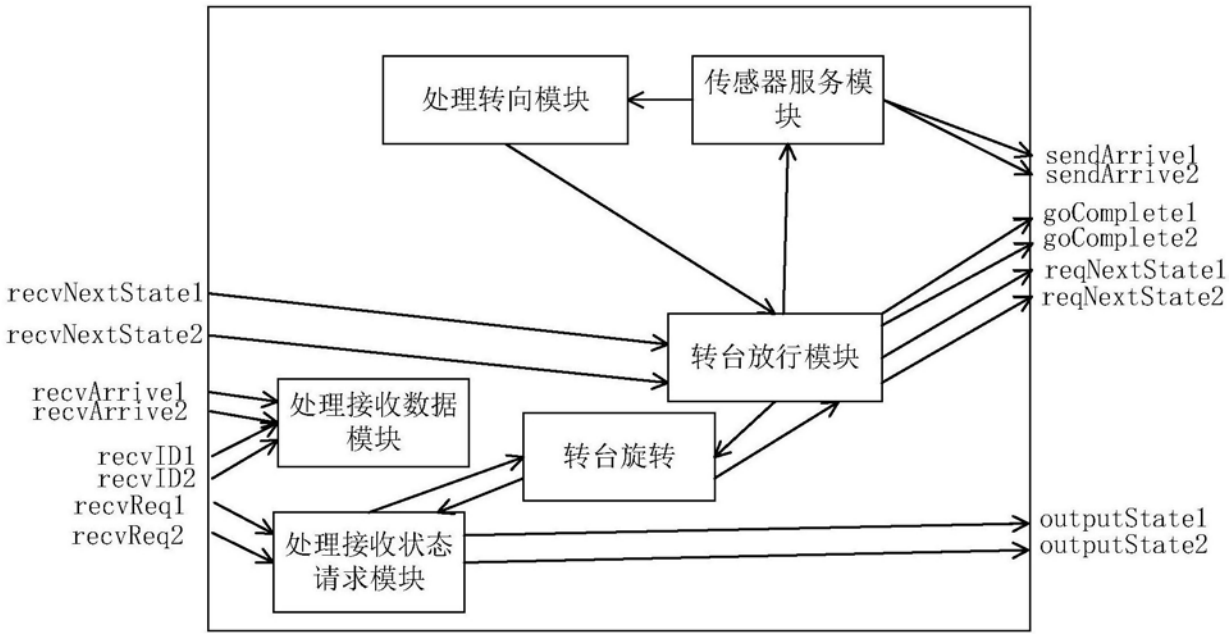

处理转向模块
传感器服务模块
sendArrive1
sendArrive2
goComplete1
goComplete2
reqNextState1
reqNextState2
recvNextState1
recvNextState2
转台放行模块
recvArrive1
recvArrive2
处理接收数据模块
recvID1
recvID2
转台旋转
recvReq1
recvReq2
处理接收状态请求模块
outputState1
outputState2

图8

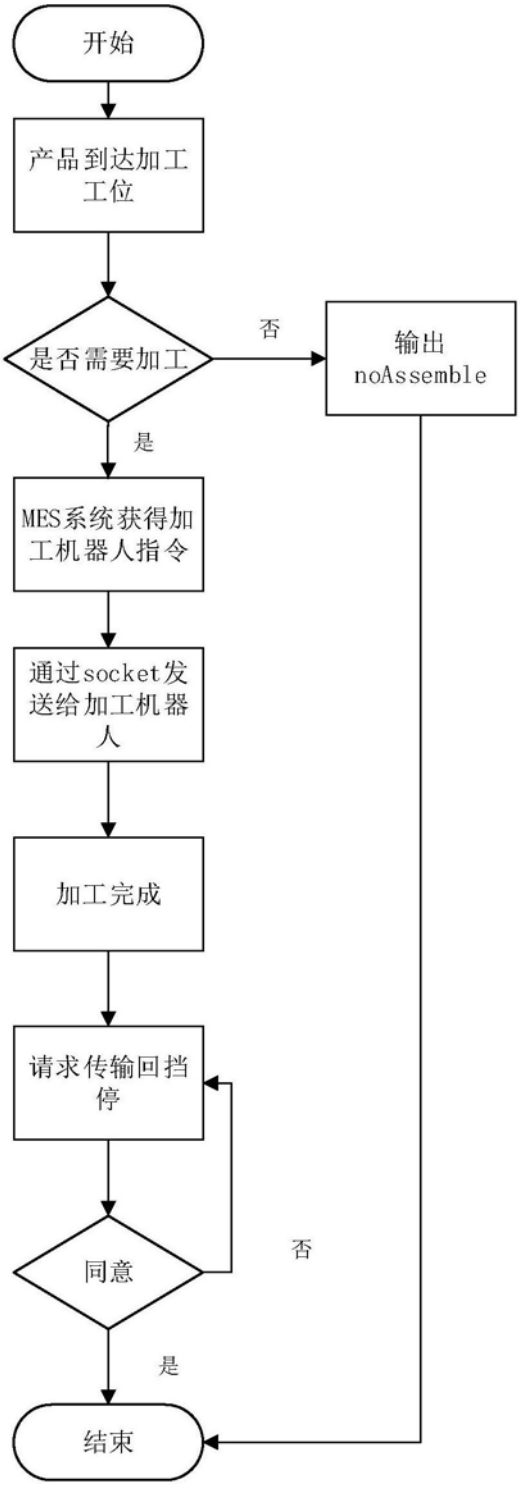

开始
产品到达加工工位
是否需要加工
否
输出noAssemble
是
MES系统获得加工机器人指令
通过socket发送给加工机器人
加工完成
请求传输回挡停
同意
否
是
结束

图9