面向复杂工业过程的通用实验平台设计与开发

王 雷 陈宗海 中国科学技术大学自动化系(230027)

Abstract

An all-purpose experiment platform for complex industry process was designed and developed based on process industrial simulation training support system.This platform comprises simulation server,operating station and building tools,running in multi-processes mode.Modeling, simulating,control and operation optimization of complex process system can be studied conveniently on this platform.The architecture and complement of the platform are introduced.Also part of codes and user interfaces of the platform software are given.

Keywords:process simulation,process control,multi-threads,pipe

摘 要

基于过程工业仿真培训支撑系统，设计并开发了面向复杂工业过程的通用型软件实验平台。实验平台包括仿真服务器、操作平台与组态工具,采用多进程的运行方式。在该平台上可以方便地进行复杂过程系统的建模、仿真、控制与操作优化研究。本文介绍实验平台的设计思想与主要实现技术,并给出部分关键代码与程序的主要运行界面。

关键词:过程仿真,过程控制,多线程,管道

连续工业过程的控制与优化研究通常不能直接在生产装置上进行,而是要经过实验研究之后才能投入实际应用。为降低实验成本并且满足反复实验的要求，有必要设计和开发相应的软件实验平台。国外已有多年的相关软件产品的开发经验,以Aspen Tech为代表的国外软件公司在石化工业中形成了垄断局面[1]。尽管如此,这些公司的竞争重点主要在通用软件方面,而通用软件的针对性不强,价格昂贵。开发具有自主知识产权的相关软件产品势在必行。我们的工作围绕面向复杂工业过程的通用型软件实验平台的研制与开发，通过对过程模拟技术与计算机仿真技术的应用研究,为软件产品化打下基础。

1 实验平台的体系结构

1.1 设计思想

通用软件实验平台(以下简称为实验平台)的设计思想来源于本实验室自主研制开发的第二代通用型过程仿真培训系统开发环境——过程工业仿真培训支撑系统[2]（以下简称为支撑系统)。

为满足多人、多系统同时培训的需要,支撑系统在硬件结构上采用星型总线拓扑,通过HUB把多台微机连接成一个局域以太网。为实现多机异地进程之间的通信,支撑系统在TCP/IP协议的基础上开发了网络底层支持软件，构成对各功能模块均透明的高速数据通道。软件上,该系统使用了结构化的设计和功能分解技术,开发出了若干相互作用、相互联系的功能模块。其中包括DCS控制单元控制算法组态软件、工艺数学模型组态软件、智能导师站软件包、工程师站软件包、现场操作站软件包、DCS操作站软件包,实现的功能十分丰富。

对实验平台来说,软件功能要求有很大的不同,因此需要采用不同的设计思想。作为用于过程建模与过程控制等应用目的的实验平台，首先要求能够方便地进行模型调试与控制方案组态。由于实验研究在同一时间段内通常仅仅由单人对单系统进行,为了使用方便,所有应用程序设计在本地运行,即不使用局域网系统,而是单机运行。这种系统构成方式也称为最小模式的运行系统。其次,程序既要能够高效地运行,又应该有良好的用户界面,所以在功能模块的设计上仍借鉴支撑系统的分离技术,即将应用程序与组态数据分离,过程模型与控制模块分离,算法模块与用户环境模块分离。

此外,实验平台软件设计运行在当前流行的操作系统上,即32位Windows操作系统,目前适用的版本包括Windows NT、Windows 2000和Windows XP。

1.2 实验平台的软件结构

实验平台由四个应用程序组成——仿真服务器程序（以下简称为服务器)、仿真操作站程序(以下简称为操作站)、算法组态程序与流程图组态程序。其中服务器程序与操作站程序组成实验平台的运行系统，而算法组态与流程图组态程序一般用于组建实验系统,在实验过程中可以不必运行。

在Win32操作系统中,每一个应用程序的运行实例称为一个进程。实验平台运行时,首先启动一个服务器进程,然后由服务器启动操作站进程,操作站进程可以不止一个,也即运行时可以包含两个以上的进程,进程间通过管道进行通信。实验平台运行时的软件结构如图1所示。

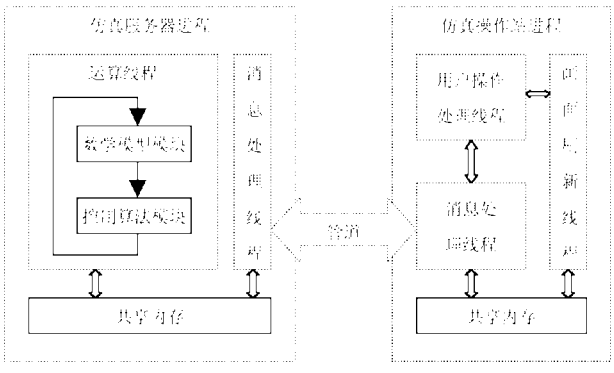

图1 实验平台运行时的体系结构图

从图1中可以看出，为保证在运行过程中正确地区分与处理系统消息和用户自定义消息，服务器进程和操作站进程中都使用了多个线程进行消息的处理。

2 数据结构与功能模块

实验平台软件实现了多种功能,数据结构的设计相当复杂,不可能一一赘述。以下将对服务器程序和操作站程序中最关键的数据结构与功能模块进行简要介绍，以便进一步阐述相关的实现技术。

2.1 数据结构

实验平台运行时包含多个进程与线程，各进程与线程之间需要频繁大量地交换数据，因此，数据结构的设计好坏直接关系到程序的运行效率。针对主要的数据应用要求，设计了三类数据结构：过程实时数据、过程历史数据、实时通讯数据。

2.1.1 过程实时数据

过程实时数据记录了所有运行时的过程仪表测量值、设定值、输出值以及设备运行状态值。它是所有其它数据的最终来源，定义为结构数组形式：

```
typedef struct {
        SINT16pointnum;                          //仪表数
    SINT16    typeno[MAXPOINT];                  //仪表索引号
    DCS_data_type    pnttag[MAXPOINT];           //仪表基本类型
    } pnttaglist_t;
pnttaglist_t          pnttaglist[MAXSTATION];
```

2.1.2 实时通信数据

实时通信数据用于服务器进程与操作站进程之间的数据通信。由于两进程每周期(一般为 1s)就要通信一次，为减小通信开销，不仅要压缩数据格式，而且通常只包含当前显示的工艺画面所涉及的过程量。具体数据内容仍然是带有索引的过程仪表测量值、设定值、输出值及设备状态值，定义为下面的结构数组：

```
typedef struct {
    BYTE      lop;                    //仪表类型
    BYTE      alm;                    //仪表报警类型
    SINT16    pv;                     //仪表测量值
    SINT16    sv;                     //仪表设定值
    SINT16    mv;                     //仪表输出值
    }   point_data_t;
typedef struct  {
    point_data_t     point[MAXPOINT];
    } control_station_data_t;
```

2.1.3 过程历史数据

在进行实验研究时，过程特性分析、建模与控制优化等都需要依据所采集的历史数据。为使用的方便，过程历史数据设计成多重格式，定义如下：

```
typedef struct {
    DCS_data_type  point;          //仪表基本类型
    short          trdtype;        //历史趋势类型
    short          trdtime;        //数据采样周期
} Trd_base_type;
typedef struct {
    char           tname[8];       //历史趋势组名
    Trd_base_type  trdpnt[6];      //历史趋势点
} trd_type;
```

2.2 主要功能模块

2.2.1 仿真服务器

仿真服务器包括以下主要功能模块：

数学模型模块：这是一个标准嵌入模块，其中的数学模型可以手工编制，也可以由组态工具生成，只要根据提供的标准接口编写，就可以直接嵌入服务器程序。

控制算法模块：另一个标准嵌入模块，任何控制算法类，只要按照标准的程序接口编写，就可以直接放入算法库，被其它模块调用。

运行状态操作模块：服务器的管理模块之一，可以完成当前运行状态的存储、过去运行状态的恢复等运行状态的管理操作。

消息处理模块：服务器的管理模块之一，区分和处理用户自定义消息以及系统消息，协调各线程的运行。

仿真时间处理模块：显示与控制仿真时间和仿真时标。

趋势数据记录模块：分层次记录过程历史趋势数据。

通信服务模块：检测操作站运行状态，通过管道向操作站发送过程实时数据、接受操作站的操作信息。

2.2.2 仿真操作站

仿真操作站包括以下主要功能模块：

流程图显示与操作模块：动态显示工艺流程图及其过程数据，响应用户在图上的键盘与鼠标操作，产生用户操作信息。

仪表显示与操作模块：动态显示被选择的仪表及其实时数据，响应用户对仪表的操作并产生操作信息。

消息处理模块：操作站的管理模块之一，区分用户自定义消息及系统消息，调用相应的处理方法。

客户通信模块：接收过程数据时，定时检测管道，按字节流方式读取管道数据并进行转换；发送操作信息时，检测管道状态为空后发送消息到服务器进程。

3 实现技术

虽然同样采用了多进程的运行机制，但由于实验平台的运行模式与支撑系统不同，进程之间不能再使用原有的 socket 技术进行通信。Win32 API 中的管道函数是实现本地进程间通信最简单而有效的技术。

使用管道通信时，需要利用 Windows 系统的消息处理机制。在服务器上，消息的处理不能影响模型与算法的计算；而在操作站上，消息的处理不能影响对用户操作的实时响应。解决这一问题的最有效方法是采用多线程技术。

3.1 多线程技术

实验平台运行在 32 位 Windows 操作系统上，这是多任务的操作系统，能将可利用的 CPU 时间分配给各个线程。系统在各个竞争的线程间分配时间，当前线程的时间完毕，它就被挂起，并允许另一线程运行。这种抢先式调度方式保证优先级高的线程首先获得 CPU 运行时间。

实验平台运行时，操作站进程的主线程产生消息处理线程、用户操作处理线程与画面动态刷新线程等多个线程。而在服务器进程中，数学模型模块和控制算法模块运行时构成一个整体，设计在同一个运行线程中；窗口对鼠标、键盘事件等消息的处理设计为另一个线程。当消息处理线程处理用户接口(键盘或鼠标的响应)时，其它线程则以较低优先级执行处理工作。

在 Visual Basic 中可以利用控件由开发环境自动建立新的线程，而在 Visual C++中可以通过调用相关的 Win32 API 函数创建线程，也可以通过调用 SetTimer 等函数由开发环境自动建立一个新的线程。详尽的实现过程可以参看文献[3]。

3.2 管道技术

Win32 系统中进程间通信的最理想方式是内存映射文件。然而，由于 Visual Basic 事实上不支持指针型变量，迄今为止尚无可靠的技术能够真正实现内存共享。Win32 API 中的管道函数为异种进程间的数据通信提供了另外一条可靠的实现途径。

在管道操作中，一方提供数据，称为服务端；另一方接收数据，称为客户端。管道有两种类型，即匿名管道和命名管道，都能够以比特流的形式传送任意数量的数据。由于实验系统是在本地运行，为使实现技术尽可能简化，我们仅仅使用了匿名管道。

在管道操作中，通常要涉及两个进程，即客户进程和服务进程。其中服务进程负责建立管道，而客户进程连接到管道。服务进程可以建立一个管道的多个实例，因此支持多个客户进程。由于管道是一块数据缓冲区，所以服务进程不必等待客户进程处理数据后才返回，同时，服务进程也可以从管道中读取数据。管

道使用内部缓存,用来在传送数据期间缓冲数据。当一个进程向管道内写入数据时，数据通常写到这个缓存内，写操作立即返回,其它进程会在方便的时候从管道中读取数据。

建立管道后,需要将句柄传送给操作站进程,实验平台中使用程序默认的标准输入和标准输出句柄。将操作站进程的管道句柄设定为标准句柄后，即实现了服务器程序和操作站进程之间的彼此通信。

实验平台的通信是双向的，即仿真服务器和仿真操作站都可以作为服务端或者客户端。进程间通信的内容是内存中的数据,在 Win32 中可以把内存当作文件一样进行操作,调用的也是文件操作函数。

从服务器到操作站传送的是过程实时数据。服务器首先定时(周期 1s 或 2s)检测是否有操作站在运行,当有操作站处于运行状态时,服务器才执行发送指令。为保证通信的准确性,先将数据长度等基本信息写入管道，再发送数据，两步都是调用 WriteFile 函数。在操作站端则利用 Windows 的消息处理机制判断是否需要接收数据，为保证正确区分和处理系统信息与用户自定义信息,调用了消息处理函数 CallWindowProc。当没有用户自定义消息时,程序正常响应操作系统的消息,如键盘输入、鼠标操作等。有用户自定义消息时,表示可以接收数据。接收数据也分为两步,先获取数据的基本信息,再根据信息接收过程数据。具体操作时,先用 PeekFile 函数检测管道中是否有数据,再通过 ReadFile 函数取出管道数据。

从操作站到服务器传送的是用户操作信息。在操作站运行过程中,当有用户操作时,如改变阀门开度、启停泵、调整控制器参数等,需要实时发送到服务器,模型根据这些参数运算给出正确的过程数据。操作站向服务器发送数据前，先要使用 FindWindow 函数定位服务器主窗口，再利用管道发送实时操作数据。为使服务器端能正确处理操作站发送的信息,同时也能正常响应 Windows 操作系统的消息,同样调用消息处理函数截获发往程序的所有消息。对用户自定义消息调用 WndProc 函数单独处理,而对 Windows 操作系统的鼠标、键盘等消息则返回调用标准消息处理过程 CallWindowProc。

4 运行画面

服务器主要完成运算功能,一般可以后台运行,所以不需要复杂的用户界面。不过为便于管理系统,仍然提供了较为方便的按钮操作功能和简洁的信息提示。运行画面如图 2 所示,主要的操作按钮包括启动仿真平台、运行状态的存储与调出等,还可以方便地改变模型运行速度。

操作站是主要的用户交互界面,运行画面如图 3 所示。其外观借鉴了仿真培训系统的仿 DCS 操作站,并根据实验平台的实际需要做了一定的修改。

图 2 仿真服务器的运行画面

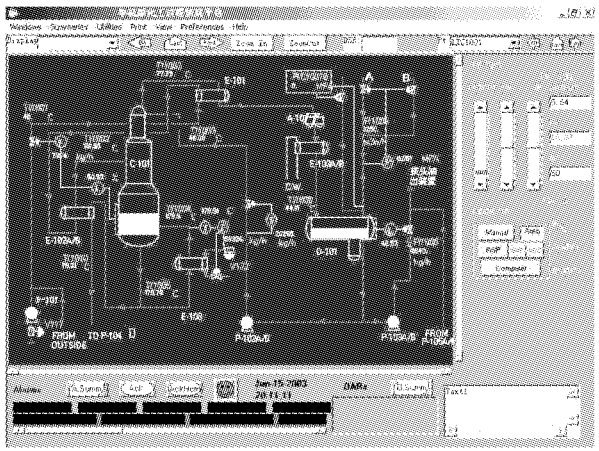

图 3 仿真槽作站的运行画面

5 结束语

该平台已研制开发成功，并在其上进行了连续重整装置预处理单元的建模与控制研究。应用效果证明了其不仅在技术上较为先进,而且使用方便。系统可以运行在最流行的硬件配置与操作系统下,为将来进一步实现软件的产品化打下良好基础。

参考文献

1 徐用懋,杨尔辅.石油化工流程模拟、先进控制与过程优化技术的现状与展望 [J].工业控制计算机,2001,14(9):21~27

2 王雷.企业级仿真培训支撑系统设计与开发[D].合肥:中国科学技术大学,1999

3 齐舒创作室.Visual C++ 6.0 开发技巧及实例剖析[M].北京:清华大学出版社,1999

[收稿日期:2003.11.4]