第29卷　第17期　　计　算　机　工　程　　2003年10月
Vol.29　№ 17　　Computer Engineering　　October 2003

·基金项目论文·　文章编号：1000—3428(2003)17 —0070—03　文献标识码：A　中图分类号：TP311

LAMOST总控模拟系统中的分布式通信机制

邹志强，徐欣圻
（中科院国家天文台南京天文光学技术研究所，南京 210042）

摘　要：结合国家重大科学工程项目 LAMOST 望远镜总控1-级模拟系统研制开发的实践，着重描述了分布式、多操作系统平台上通过TCP/IP、IPC（进程间通信）来协调完成多个分布式任务的通信机制。并以此机制为基础，对望远镜机架和焦面控制进行了很好的模拟。

关键词：虚拟望远镜；模拟控制系统；QNX；Linux；TCP/IP；进程间通信

Distributed Communication Mechanism of LAMOST Simulating Control System

ZOU Zhiqiang, XU Xinqi
(Nanjing Institute of Astronomical Optics & Technology，National Astronomical Observatories of Chinese Academy of Sciences，Nanjing 210042)

【**Abstract**】With the practice of the national impontant scientific project of large sky area multi-object fiber spectroscopic telescope (LAMOST), this paper presents the mechanism how to complete the multi-tasks which are distributed on multi-OS (operating systems) by means of TCP/IP and IPC(inter process communication). Furthermore, based on this mechanism, the simulating control of the telescope mounting rack and focal plane is implemented well.

【**Key words**】Virtual telescope; Simulating control system; QNX; Linux; TCP/IP; IPC

1 概述

LAMOST是我国“九五”重大科学工程，项目投资2.35亿元。建成后它将成为世界上4m级口径以上光学望远镜中视场最大和光谱观测效率最高的望远镜。这样大的工程不仅给光学和机械的设计提出了很高的要求，也给望远镜控制系统的设计提出了挑战。控制系统的基本功能在于给LAMOST望远镜的各种信息，如命令信息、数据信息、状态信息等，提供通道。根据总体设计的要求以一定的规律和精度通过各电控子系统驱动望远镜的电控运动部件，与光机协调完成对目标天体的观测任务。LAMOST控制软件的研制，是一个复杂而又庞大的软件工程。由于在研制过程中各被控实体不可能一下子到位，因此必须对各被控实体的控制进行模拟。

1.1 LAMOST虚拟望远镜

定义：LAMOST望远镜总控模拟系统，或者可称为LAMOST虚拟望远镜，是一个运行在本地网络上的复杂的软件包，它包含大量不同层次的模块，此外还有人机接口。这些模块和接口根据一定的规则协调运行，以模拟该望远镜各种总控命令的运行。除了本地网络硬件以外，被控实体并不参与模拟运行。

1.2 具体的被控对象

与LAMOST望远镜控制系统TCS(Telescope Control System)相关的对象，有上层的观测控制系统OCS(Observatory Control System)和下层的8个子系统（望远镜机架控制子系统,焦面控制子系统，主动光学控制子系统等）。如何来模拟它们的各自功能，并协调工作，这就是我们的任务。

1.3 对象的模拟实现平台

硬件上，TCS用一台Compaq AP500来模拟，其键盘和显示器兼作控制台。其它3台NEC PIII500分别模拟观测控制系统OCS、机架控制子系统MRCS (Mounting Rack Control System)和焦面控制子系统FPCS(Focal Plane Control System)等。

软件上，LAMOST望远镜总控1-级模拟系统非实时部分采用Linux/Unix操作系统（Red Hat 6.2），实时部分采用QNX操作系统（QNX 4.25）。QNX作分布式应用时，使用的是自己的内部协议(Fleet Protocol)，TCS及其子系统之间的通信由IPC(进程间通信)来完成。TCS和OCS之间的通信通过TCP/IP协议来实现。

QNX系统上的控制软件开发用Watcom C/C++，用户界面的编程使用QNX软件包中Photon microGUI所提供的工具。QNX操作系统的主要特点有：多任务多用户；分布式。从用户角度看，QNX局域网就像一台集中式的多用户计算机，在QNX局域网上的每个用户都可以透明地使用该网络上他有权使用的任何资源。QNX是一个实时操作系统。它提供用户可控制的、优先级驱动的、急者优先抢占的调度方法。它符合Posix基本标准和实时标准。

2 模拟系统中OCS与TCS接口间数据流的描述

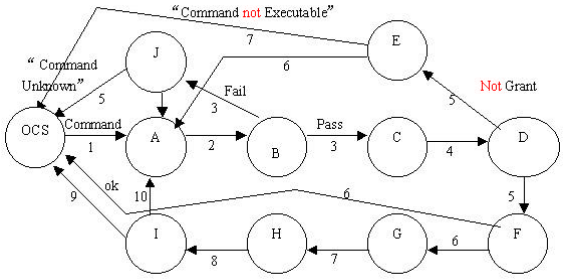

图1 OCS和TCS数据流图

基金项目：国家“九五”重大科学工程基金资助项目
作者简介：邹志强（1967—），男，工程师、硕士生，主要研究方向：计算机控制；徐欣圻，研究员、博导
收稿日期：2002-09-24

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

A: Command Receive，命令的接收；B: Grammar Check，语法的检查；

C: Command Scan，命令的扫描；D: Execution Condition Check，执行条件的检查；

E: Check Fail，检查失败；F: Check Success，检查成功；

G: Command Execution，命令的执行；H: Object Control，对象的控制；

I: Status Check，状态的检查；J: Grammar Check Fail，语法的检查失败；

图1是我们执行任务的数据流图，其中OCS（观测控制系统）的软件平台是Linux,而状态AJ 是由在TCS(望远镜控制系统)上的多个进程来完成的，TCS的软件平台是QNX(类Unix)。下面介绍如何通过TCP/IP、IPC来协调完成这些分布在多个操作系统上的多个任务 。

3 模拟系统的实现

3.1 模拟系统的功能图

先按照功能进行合并，把功能类似的合并为一组，如图2所示，每一组由一个进程来完成。OCS和TCS之间的联系通过TCP/IP协议由Socket通信来完成，TCS内部各个进程之间的联系通过Fleet协议由IPC来完成。

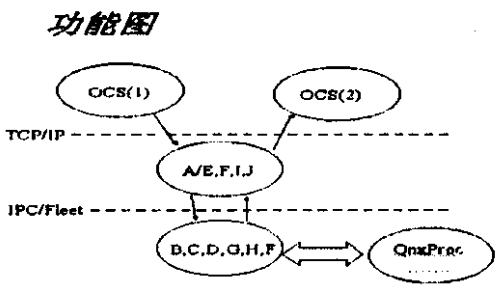

图2 功能图

3.2 模拟系统的逻辑图

相对于功能图，我们在实现时采用的是客户/服务器模式（C/S），但由于功能上的需要，采用了双重C/S模式，具体如图3所示。

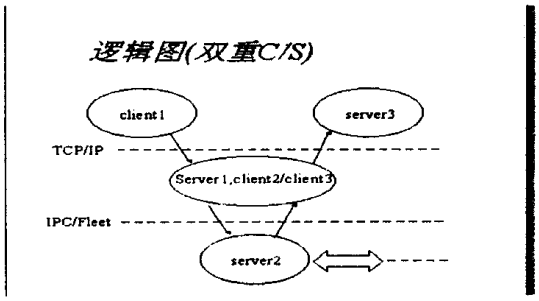

图3 逻辑图

3.3 构建模拟系统的框架

3.3.1 Linux下望远镜观测控制系统的模拟

由于各个控制子系统的模拟都是基于通信的，因此在Linux下要有一个客户进程，负责发送各种命令。要实现的功能是功能图中的OCS(1),相应的程序为lclient.c，具体的命令格式如下：

<command>

<command> <parameter1> <parameter2> ...<parametern>

(1) Some examples of commands:

TcsFullyAutomatic

TcsDatabaseAccess

(2) Some examples of command parameters

CenterDeclination*<24d54m11s>*

<GuidingStar 44d23m18s 23d55m54s>

这些命令和参数是通过在命令行输入文本执行的。例如:

lclient StartObserve*GuidingStar 44d23m18s 23d55m54s*

由图2可知，命令是在Linux下发出的，而命令的接收是在QNX（类Unix）下进行的。所以，采用TCP/IP协议，通过socket通信来完成。为此，先定义如下数据结构：

```
typedef struct {
short unsigned SubSysName;//子系统的名字
char StatusName[16];//状态信息
short unsigned orderno;//顺序号
time_t crnt_time;//状态量采集的时间
float value[2];//状态数据的值
short unsigned onoff;// 状态数据的开关量
char remark[8];//正常与否的说明
}g_status_t ; //全局的状态类型，与TCS_L1中的类型一致
g_status_t status_from_tcs;
```

完成发送命令的函数如下：

socket(AF_INET,SOCK_STREAM,0),gethostbyname(TCSHOST), write(sock,status_from_tcs, sizeof(status_from_tcs))...。

3.3.2 QNX下的服务器端的模拟

由图3可知，在QNX端需要一个与3.3.1中lclient.c 相对应的服务器程序，并且这个程序还要负责与QNX端的其它进程交互信息。因此把它称为桥梁进程bridge，对上层它采用socket通信，对下层它采用IPC通信。进程bridge是一个守护进程，其角色相当于逻辑图中的server1,守护等待Linux上传来的消息；并把此消息传给QNX 上的界面进程如TCS-L1，其角色相当于逻辑图中的client2。值得注意的是，此时，进程bridge 和TCS_L1同在QNX操作系统下，进程TCS-L1要想正确接收到进程bridge的消息，也必须处于守护状态，其角色相当于逻辑图中的server2。再由TCS_L1经命令扫描程序后发给各个子系统，如MRCS(机架控制系统), FPCS(焦面控制系统)等。各个子系统处理完相应的命令后，把执行的结果返回给进程TCS-L1,最后经进程bridge把结果传给OCS(观测控制系统)，此时进程bridge的角色相当于逻辑图中的client3,与此对应在Linux下的服务器进程是server3。由此，可以看出这是一个双重的客户服务器模式，其中bridge是一个关键进程。

3.3.3 QNX下望远镜控制系统的模拟

因为，进程bridge 和TCS_L1同在QNX操作系统下，所以我们采用Fleet协议，通过IPC来完成信息的交换。为此，先定义如下数据结构：

```
struct msg_to_subsystem {
short          unsigned type;        //子系统的名字
pid_t   sendpid;   //发送者的进程号
char    string[81];          //具体的内容
int     action;    //执行动作的名称如：STARTUP, TRACKING
double  data[3];             //[0]:方位; [1]:高度;[2]:速度
 };
struct msg_subsystem_reply {
int     status;    //子系统的状态
char    string[81]; //具体的内容
};// Define the union of all messages
```


© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

```
typedef union {
struct msg_to_subsystem    to_subsystem;
struct msg_subsystem_reply subsystem_reply;
}pg_tcs_msg_t;   //整个项目公用的消息结构
pg_tcs_msg_t cmd,replyCmd;
```

然后，对需要及时回应的消息采用同步通信方式（具体的函数如下），对不需要及时回应的消息采用异步通信方式（参见3.3.5）。

```
Send( pid, &cmd,&replyCmd,sizeof( cmd),sizeof(replyCmd) )；
input_ID = PtAppAddInput ( NULL,0, RevMessage , NULL);
//加入事件空间。
```

注意这里是一个标准控制台的守护进程和一个用PHAB开发的图形界面进程之间的通信。所以，不直接采用Receive(pid1, &msg, sizeof(msg))，而是采用PtAppAddInput( )。把接收消息的函数加入QNX－Photon的事件空间，把等待的任务交给系统由系统去统一处理，这样可以继续做其它事，而不影响图形界面的刷新。具体的接收消息函数如下：

```
//从bridge进程接收数据
int RevMessage (void *client_data, pid_t pid, pg_tcs_msg_t *cmd,size_t cmdlen) //对发送进程的回答
Reply( pid, &replyCmd, nbytes );
```

在消息被接收到后，对消息进行分类和检查，如图1中数据流图中的B: Grammar Check, 语法的检查；C: Command Scan，命令的扫描；D: Execution Condition Check，执行条件的检查；具体函数如下：

```
//处理来自bridge进程的字符串信息的函数
void StringProc( char buf[CMD_LENGTH])
void CommandProc (int iCmd,char rev_para[5][20]) //命令处理函数
```

对检查合格的命令，也要通过IPC（进程间通信）送给另外的进程，如MRCS_L1, FPCS_L1, AOCS_L1。那么，首先要知道对方的进程号，才能发消息给它。在QNX中通过系统调用:pid=qnx_name_locate(4,"MRCS_L1",0,NULL)；来得到对方的进程号。如果得到的消息是命令“Start-Observe”,那么我们还要进行科学计算，把计算结果发送给进程MRCS_L1。其中计算是指：把要观测星的赤纬和时角换算成望远镜的机架的高度和方位。此时，发送进程是在1号节点上的进程TCS_L1，而接收进程则是在2号或3号节点上的进程MRCS_L1，并且要求:进程TCS_L1从得到命令，科学计算,到进程MRCS_L1收到消息,此段时间要小于20ms。实验表明我们的数据可以满足需要(见第4节）。

以上讨论的是：进程TCS_L1对来自上层进程OCS的消息处理，同时进程TCS_L1也担负着对下层各子系统的联络。例如，在系统维护和调试时，进程TCS_L1应能向下层各子系统发送各种命令，并把执行结果返回给上层进程OCS。我们对各控制子对象进行分析，提出了LAMOST 1级模拟系统中的命令分类表,并对语法进行了规定,具体如下：

数据库访问命令表;机架控制命令表;主动光学控制命令表；焦面控制命令表；系统状态检查命令表等。每一个表都是一个命令的集合，例如，在机架控制命令表中有：

TcsPointingTo：TCS机架指向命令。其语法：参数可以是两个，即方位值和高度值。当两者中缺一时，望远镜将单轴旋转到参数值的位置。当两个参数都缺省时，望远镜不动。参数也可以是3个值，即目标星的方位、高度和相应的时间值。命令执行的结果是，在该时间值，望远镜机架运动到参数指定的位置。

TcsStartTracking：TCS启动跟踪命令。其语法：该命令必须带有被跟踪天体的赤经参数和赤纬参数。

TcsStopTracking：TCS关闭跟踪命令。其语法：该命令不带参数。通知进程MRCS_L1停止PID等算法的执行。

3.3.4 子系统的模拟

各个子系统主要是通过分布在QNX各节点上的不同进程来模拟的。例如：node2上的进程MRCS_L1来模拟望远镜机架的方位和高度控制子系统,进程FPCS_L1来模拟焦面控制子系统;node3上的进程AOCS_L1来模拟主动光学控制子系统。其中，进程MRCS_L1一旦收到进程TCS_L1发来的消息：TcsPointingTo，则启动定时器中断。在中断程序中，根据指定的目标位置和当前的位置，以及通过实验得到的比例P，积分I，微分D3个系数来执行PID算法，得到机架的运动速度，并让机架以此速度指向目标位置。

如果当前位置与目标位置之差小于某一设定值，则自动运行跟踪算法，并记录跟踪误差。具体实验数据参见第4节。为了让用户操作更方便，在PHAB下，开发了动态的图形界面，随时显示指向和跟踪的位置。

其它子系统的模拟与机架的控制子系统类似，我们的宗旨是：在执行上要满足实时性，在界面上要具有友好性。

3.3.5 LAMOST总控模拟1级系统中的异步通信

在3.3.3中讨论了同步通信，即在时间轴上，一个进程A在发送消息到进程B之后，必须等到进程B的回答才能进行进程A下面的工作。现在简单介绍一下对实时性不强的消息采用的异步通信。利用Photon中的Pulse和signal方式，即此时，进程A在发送消息到进程B之后，不必等到进程B的回答就可以执行进程A下面的工作。

对于Pulse发送者，我们要：(1)在Photon的事件空间中注册一个句柄，以便在事件发生时由系统通知该进程，从而调用相应的事件处理函数。(2) 触发一个Pulse,PtPulseDeliver(pid_t rcvid, PtPulseMsg_t const &pulse_msg)，其中rcvid是接收者的进程号，pulse_msg是与接收者中的结构相一致的。

对于Pulse接收者，我们要：(1)创建Pulse；(2)绑定Pulse，PtPulseArmPid (NULL, pulse,pu lse_deliverer,&pulse_msg)；(3)与发送者联络；(4)注册一个句柄和相应的事件处理函数；(5)在不用Pulse时，解除绑定，释放资源。其中pulse_deliverer是Pulse发送者的进程号。

4 分析和总结

在上述各软件模块都编制完后，我们设置了一个启动进程StartProj来向系统创建上述各个进程，以满足系统自动模拟的需要。

在3.3.3中，我们对赤纬从-10°~+90°的500组数据进行计算，以一般用户权限，进程优先级设为19，此时平均耗时为150.6ms，对一组数据而言时间为0.3012ms。数据在TCS_L1进程和MRCS_L1进程之间的交互时间约为2ms[4]，所以总的耗费时间小于3ms,能满足设计的需要。

在3.3.4中，MRCS_L1进程为确定机架的位置，采用了PID算法。当初始位置为10，目标位置为50，比例P，积分I，微分D3个系数分别为：0.8，0.65，0.1，经32步后，指向误差为：0.114;在开始跟踪后，前10步的跟踪误差平均为：0.08502，数字调节较好，可以满足模拟的需要。

整个模拟是在类Unix的系统QNX下通过大约8千多行C语言代码完成的。现在，我们已经按计划，构建了LAMOST望远镜总控模拟系统的基本框架，尤其是对望远镜机架控制和焦面控制进行了很好地模拟。但是，随着开发的进展，系统还要提供各种被控硬件的接口，我们仍会继续努力，把将来的工作做得更好。

(下转第92页)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

量机算法、神经网络方法、最大平均熵方法、最近 K 邻居方法和贝叶斯方法等。

在本系统模型中使用基于向量空间的简单向量距离分类方法是基于以下考虑的，文本在向量化之后就表现为向量的形式，这个过程是和语种相关的，如中文、日文需要分词，英文需要进行形态变化。而此后的过程都是和语种无关的，这就为研究独立于语种的文本分类方法提供了可能性；并且这一分类方法思路十分简单，分类花费的时间较短，更利于文本的实时处理。具体步骤如下：

（1）计算每类文本集的中心向量；计算方法为所有训练文本向量简单的算术平均。

（2）新文本到来后将文本表示为特征向量。

（3）计算新文本特征向量和每类中心向量间的相似度，公式为

$$\mathrm{Sim}(d_i, d_j) = \frac{\sum_{k=1}^{M} W_{ik} \times W_{jk}}{\sqrt{(\sum_{k=1}^{M} W_{ik}^2)(\sum_{k=1}^{M} W_{jk}^2)}}$$

其中，d_i 为新文本的特征向量，d_j 为第 j 类的中心向量，M 为特征向量的维数，W_k 为向量的第 K 维。

（4）比较每类中心向量与新文本的相似度，将文本分到相似度最大的那个类别中。

3.4 层次分类

对于某个对象，分类器对应每个类别进行匹配。都要计算出一个数值来表示该对象属于该类别的程度。本文分类系统用文档和类别所表示成的空间向量之间的相似度来表示两者之间的相似度。若相似度大于对应阈值，则分类成功。

在进行特征提取时，一般的做法是将所有文档类都视为同一层次。提取出各类在同一层次条件下的特征矢量。在分类时，再计算待分类文档与各类的相似度。如满足阈值条件则将其划分到相应的文档类。当文档类较少并且文档类之间的区分度较大时，这种匹配策略能够较为有效地进行文档类的划分；但当文档类的数目较多，且有部分类之间的相似度较大时，利用上述的匹配策略就很难区分内容较为相似的文档。并且在分类时需要进行所有文档类的匹配计算。

为了提高对类别的区分能力和分类速度，对匹配策略进行如下的改进:将文档类划分为树状的多个层次，内容相近的类组合成一个大类，内容相似的大类再组成更上一层次的大类。并计算出每一层次类的特征向量。在进行相似度匹配时：根据文档类的层次从上至下一层层地匹配。本系统是根据问题库中的问题类型来对文档进行类别层次划分的。例如，对位置提问的相关文档类，对时间提问的相关类，对人物、对数量、对定义等提问的相关文档，同一主题的这些类将构成这一主题大类的子类。采用上述匹配策略：一方面由于将文档类进行了划分和分层，使得各文档之间的区分特征变得明显了，有助于提高分类的准确度；另一方面，由于进行了分层匹配，大大减少了相似度计算和匹配的次数，因此能够进一步提高分类的速度。

4 结束语

随着因特网的不断普及和发展，各类电子可读文本日益增多，文本分类是处理海量文本的良好手段。本文给出的分类方法的特点是：考察了网页中的特殊描述信息，并对其做相应的加权处理；运用可分性判据和层次分类来增加信息的可分性；并且利用对问题库的浅层语义分析提高分类的准确度。应该说这种改进后的分类方法在相似文本分类中具有明显的优势。在以后的研究中，将研究如何再提高分类的效率，以满足对文本的及时分类；此外，考虑加入反馈功能，即对没有正确分类的文本，将其作用到训练文本，使分类器能对该文本及其相似文本正确分类，这是符合信息过滤的分类思想的。

目前，正在着手进一步的后续工作，完善该方法并做进一步的试验，并与传统的分类方法比较。初步的工作表明，该方法是行之有效的。

参考文献

1 Koller D. Hierarchically Classifying Documents Using Very Few Words. Proceedings of the Fourteenth International Conference on Machine Learning (ICML-97), 1997

2 Zhang Li, Li Xing. Net-compass, A Search Engine for Chinese Web Pages[A]. The First AEARU Workshop on Web Technology[C] , Kyoto, Japan, 1998:10-15.

3 孙 健. 基于K_最近距离的自动文本分类的研究. 北京邮电大学学报, 2001 , 24(1)

4 黄萱青. 独立于语种的文本分类方法. 中文信息学报 , 2000 , 14(6)

5 孙丽华. 一种改进的kNN方法及其在文本分类中的应用. 应用科技 , 2002 , 29(2)

6 朱华宇, 孙正兴, 孙福炎. 一个基于向量空间模型的中文文本自动分类系统. 计算机工程 , 2001 , 27(2)：15

7 周水庚. 隐含语义索引及其在中文文本处理中的应用研究. 小型微 型计算机系统 ,2001 , 22(2)

8 解冲锋. 补偿型的Sleeping expert文本分类算法. 清华大学学报 , 2001 , 41(7)

9 庞剑锋. 基于向量空间模型的文本自动分类系统的研究与实现. 计算机应用研究 , 2000 , (9)

☆☆

（上接第69页）

2 于慧龙. 基于保护轮廓的数据库系统安全性[J]. 信息网络安全, 2002,(7)

3 赖溪松, 韩 亮, 张真诚. 计算机密码学及其应用[M]. 北京：国防工业出版社, 2001

4 Encryption of Data at Rest-Database Encryption ——White Paper. http: // www. appsecinc.com

☆☆

（上接第72页）

参考文献

1 徐欣圻. LAMOST望远镜总控模拟系统 LAMOST. 工程部内部技术报告, 2001-2002

2 QNX OS System Architecture.(Second Edition). QNX Software Systems Ltd.,1999

3 周 军. LAMOST望远镜总控0级模拟系统[硕士学位论文].南京: 中科院南京天文仪器研制中心,2000

4 金刚平. LAMOST控制系统时标系统 [硕士学位论文]. 南京: 中科院南京天文光学研究所，2001

5 徐灵哲. LAMOST分布式数据库设计南京中科院南京天文光学研究所 ,2002

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net