第16卷 第11期
2006年11月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.11
Nov. 2006

串口通讯中系统资源分配问题的研究

江 峰,刘高嵩
(中南大学 信息科学与工程学院,湖南 长沙 410075)

摘 要:通常串口设备与主机间的通讯是一一对应的。而客户端与服务器间的通讯一般都是多线程并发的,因此在多客户端调用服务器端串口设备时就会产生矛盾。文中从软件的角度出发,探讨了用程序方式控制客户端请求,协调多客户端资源分配及数据传递的可能性。提出串口操作函数加锁、设置服务器端队列、握手协议二次开发和快速反应4种解决方案。最后分析了这4种方法在实际应用过程中各自的优缺点。

关键词:串口;加锁;队列
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2006)11-0064-03

Research on Operating System's Resource in Serial-Port's Communication

JIANG Feng, LIU Gao-song
(College of Information Science and Technology, Central South University, Changsha 410075, China)

Abstract: Usually, the communication type between server and serial-equipment is point-to-point. But the communication between client and server is multi-thread parallel running. So it arises contradictions when multi-clients want to use one serial-equipment on the server at the same time. Discusses the possibility of using program to control the client's requests and corresponding the resource distribution and data transfer between one client and another from the software side. And brings forward four solutions: locking for the serial operating function, set queue on the server, the second developing to the handshake protocol and quickly response. And then analyses the advantages and disadvantages of each other in practical situation.

Key words: serial-port; lock; queue

0 引 言

串口设备具有串行执行的特点,在实际应用过程中,当某个串口资源位于服务器端,而多客户端对其进行调用时,往往会引起资源占用冲突。

由于串口设备串行执行,因此不能及时对并发请求进行处理,会造成请求的丢失或返回错误的执行结果,串口设备本身成为系统的瓶颈。

如何既能合理地处理多客户端的并发请求,又能保证串口执行结果的正确性,使其能正确地完成所需功能,这是在实际设计时必须讨论的问题。

1 解决思想和方案

1.1 解决思路

如何正确地解决资源冲突问题,可以从硬件和软件两方面进行考虑。

从硬件方面考虑,可以在服务器端并列运行多个串口设备,或采用异步通信的方式,以提高服务器端的响应速度。

从软件方面考虑,针对服务器端串口设备的瓶颈问题,可以考虑在程序中控制和处理客户端请求:一是考虑限制请求使其串行发送,串行执行。二是考虑对并发请求进行缓冲,以适应服务器端串口设备的执行速度。

1.2 解决方案

文中主要从软件方面,对如何合理地分配管理串口资源提出了串口操作函数加锁、服务器端队列、握手协议二次开发和快速反应4种方案。

1.2.1 串口操作函数加锁方案

对服务器端串口操作函数加锁,即使用临界区信号量或锁操作。

例如在Delphi中,可以将关键操作封装成一个函数,在主程序调用该函数时,使用TCriticalSection.Enter。调用完成后,使用TCriticalSection.Out,这样,当系统中同时有多个线程调用该函数时,先调用的线程进入加锁函数,获得串口资源。而后调用的线程必须排队,等到前面线程调用结束后才能获得串口资源[1]。

该方案中,加锁函数自动对客户端的请求进行排队。其操作由系统自动完成。因为只对关键操作函数进行加

收稿日期:2006-03-02
作者简介:江 峰(1983-),男,湖南安乡人,硕士研究生,研究方向为构件技术和分布式系统;刘高嵩,副教授,研究方向为软件工程、构件技术、MIS系统。

锁就能实现对串口的正常操作,因此实现比较简单。如图1所示。

但在实际应用中,串口的关键操作需耗时若干秒,这样当有多个客户端同时请求串口资源时,最后请求资源的线程需要等待 $N-1$ 倍串口关键操作时间。这在实际应用中是不能接受的。

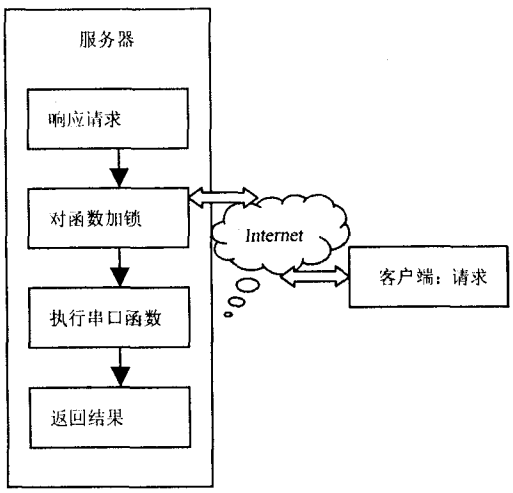

图 1　串口操作函数加锁方案程序执行顺序图

1.2.2　设置服务器端队列方案

在主程序中建立一个队列,当客户端请求串口资源时,就将该请求入队列,然后用单独的线程对队列中的请求进行处理。

对于执行结果的返回,可以规定当客户端与服务器建立 TCP 连接之后,每次调用串口的操作都建立在该连接上,等所有操作完成并得到返回结果后,再断开连接[2]。

此时需要讨论如何将串口执行结果正确地返回给客户端。提出如下解决方案。

当客户端的请求入队列后,应在有限时间内给出串口执行结果。考虑到网络稳定性因素,客户端应该同时设计有超时功能和捕获网络异常功能。服务器端每次从队列中取出一条信息(该信息应包含客户端 IP 地址及相对请求序号,请求序列号使用递增的全局变量保证唯一性)。然后调用串口关键操作。并将返回结果存储在数组中[3]。如图 2 所示。

(1)客户端可根据 IP 地址和请求序号组合,向服务器端询问执行结果。按发送先后顺序查询,在超出某个时限后,判断为超时。

(2)或者当服务器执行完一次串口关键操作后,按照信息条中 IP 地址和唯一序号将执行结果返回给客户端。如果客户端按上述时间等待无应答,则判断为超时。

该方案中,服务器端接收请求和执行串口函数是相互独立的,因此能有效地缓冲多客户端并发请求的情况。

但是,在将执行结果返回给客户端这一过程中,该解决方案要求客户端具有多线程处理能力,或者数组处理和事件监听能力,因此增加了客户端的实现难度。而且对于少请求与多请求客户端并发入队列的情况,可能会出现少请求客户端需要等待很长时间才会有返回结果。一般会造成超时,因此降低了系统可靠性。

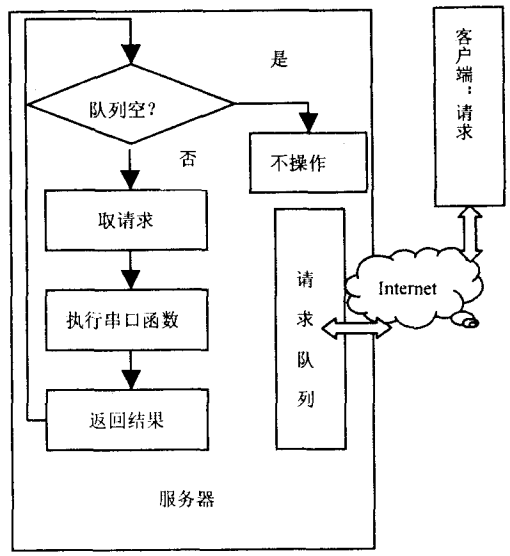

图 2　服务器端队列方案程序执行顺序图

1.2.3　握手协议二次开发方案

串口是独占资源,相当于同一时间只能由一个客户端进行调用。因此可以考虑在串口资源被占用时,对其它客户端的请求给予资源无法获取的应答,从而拒绝其调用。

初步的设想是在 TCP 三次握手协议上进行二次开发。

在具体的实现过程中,为串口资源设立一个是否被占用的变量 Locked(初始化为假)。执行过程如下[4]:

(1)客户端与服务器建立 TCP 连接。

(2)客户端询问串口资源是否可用,服务器端返回 Locked 值。

(3)如果 Locked 值为假,客户端将串口执行关键操作所需的必要参数传递给服务器。如果返回的 Locked 值为真。客户端将执行第 5 步操作。

(4)服务器将 Locked 赋值为真。调用串口关键操作。并将串口执行的结果返回给客户端。如果有多次调用,重复执行该步骤。

(5)客户端传递某一规定参数给服务器端,以表明自己是否调用过串口资源,并断开连接。

(6)服务器根据传回参数做出判断,如果调用过串口资源的话,在客户端断开连接时将 Locked 赋值为假。否则不进行操作。

程序流程如图 3 所示(注:(r)表示接收数据,(s)表示发送数据)。

这种情况下,当已有客户端占用串口资源时,如果另一客户端请求资源时,会得到被锁的提示,因而自动断开连接。

理论上该方案是一种很好的解决办法,但在实际的测试过程中,存在以下几个问题。

由于在并发环境下,多次握手操作之间可能会被插入其它操作。因此有可能出现以下情况,当某一客户端得到

串口的返回结果后，准备发送已申请关键操作标识，并与服务器端断开连接时，CPU 转向处理并发的另一客户端连接请求。因为此时系统的 Locked 属性尚未赋值为假（如果是在关键操作执行完后立即置 Locked 为假，仍然可能在上下操作之间被插入其它操作），所以另一客户端自动断开。这样服务器端断开事件中得到的是否请求过关键操作标识就是第二次的请求结果，而第一次连接中发送的标识会被丢失（服务器端读到为空）。因此没有操作将 Locked 赋值为假，串口资源无法解锁，会一直死锁下去。

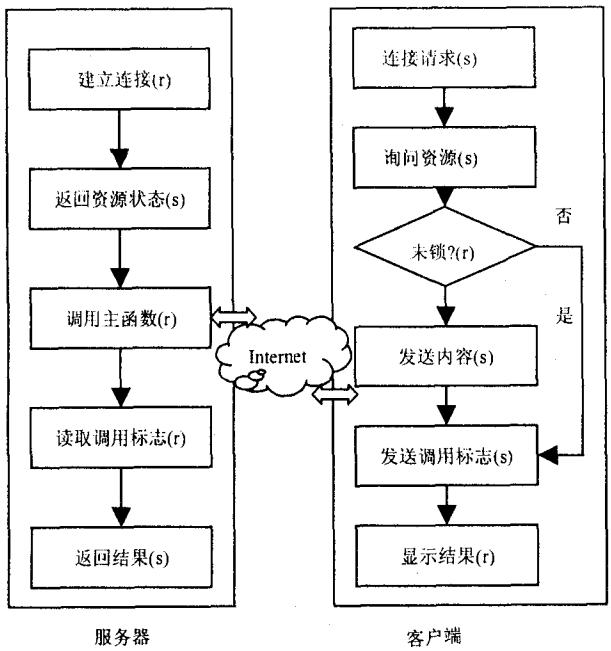

图 3 握手协议二次开发方案程序执行顺序图

1.2.4 快速反应方案

摒弃二次开发 TCP 握手过程。采用对客户端每一次申请串口资源尝试一次连接，以希望能对每次尝试连接快速给出反应[5]。

当客户端与服务器建立连接后，首先检查服务器端监听的线程数。如果线程数等于 1，则与服务器端相连的只有一个客户端，可以将串口资源分配给它。当线程数大于 1 时，直接调用断开操作。如图 4 所示。

在实际测试过程中，同样可能出现在客户端调用断开操作后，服务器端触发断开事件前，被插入其它客户端或该客户端本身的另一请求。若请求来自其它客户端，则对系统可靠性不产生影响，因为此方案中未使用 Locked 变量。若请求来自该客户端本身，则可能出现已断开连接，又要求二次断开的情况，这样系统会捕捉到访问 0 内存地址异常。解决方法是在客户端以某种方式保证在上一次请求未得到返回结果前，不触发下一次请求（例如可以将请求的 Button 设置为不可点击）。

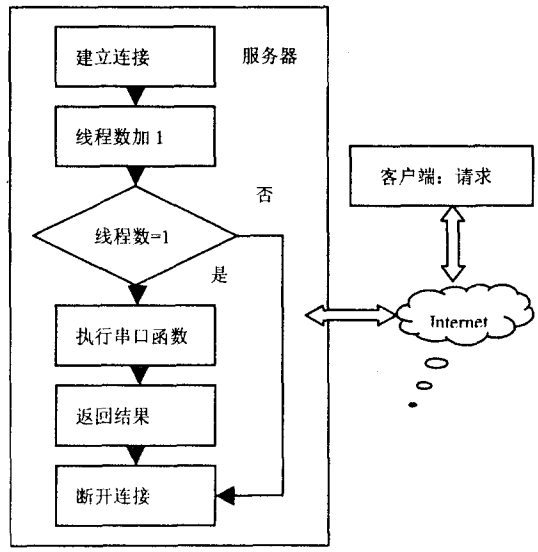

图 4 快速反应方案的程序执行顺序图

2 结束语

前两种方案适用于客户端具有多线程处理能力的情况。这种情况下，系统处于多线程并发执行状态，虽然总的执行速度和后面方案的串行执行一样，但对客户端做到了真正的即时响应。但由于交互过程比较复杂，因此系统稳定性较差，而后两种方案对于客户端要求比较低，整个系统相当于完全处于串行执行状态，因此逻辑实现比较简单，系统也相对稳定。

参考文献：

[1] 李 维. Delphi 5. x 多层分布式应用[M]. 北京：机械工业出版社，2000.

[2] 张尧学，史美林. 计算机操作系统教程（第 2 版）[M]. 北京：清华大学出版社，2000.

[3] 鲍 敏，吴 昊. Delphi 网络高级编程[M]. 北京：人民邮电出版社，2001.

[4] 微软公司. 微软管理解决方案：作业调度[EB/OL]. 2002. http://www. microsoft. com/china/technet/itsolutions/techguide/msm/smf/smfjobsc. mspx.

[5] 宗大华，宗 涛. 操作系统[M]. 北京：人民邮电出版社，2002.

（上接第 63 页）

2001.

[3] 张文修，吴伟志，梁吉业，等. 粗糙集理论与方法[M]. 北京：科学出版社，2001.

[4] 王国胤. Rough 集理论与知识获取[M]. 西安：西安交通大学出版社，2001.

[5] Sever H. Knowledge structuring for database mining and text retrieval using past optimal queries[D]. LA, USA: The university of Southwestern Louisiana, 1995.