数据队列技术在 AS/400 客户机/服务器模式中的应用

梁伟晟　陈剑鸿　（中山大学计算机系　510275）

摘要：本文讨论了数据队列技术原理，并通过一个实例分析了如何利用数据队列技术实现 AS/400 和 PC 程序间的高速数据交换。

关键词：C/S 模式　数据队列技术　有键数据队列　无键数据队列

一、引言

数据队列是 AS/400 上存放应用程序间交换数据的一种系统对象，具有可以被多个作业同时访问、无固定信息格式等特点。实际上，利用数据队列进行程序间数据交换的技术可以说是 ODBC 和 CPI－C 技术的折衷，由于它速度较快，易于理解和编程，一直都是 AS/400 客户机/服务器程序的热门技术，得到了广泛的讨论。数据队列具有以下优点：

·AS/400 上一种高速的通信方式；

·占用的系统资源少，所需设置也少；

·一个批处理程序可以只用一条数据队列为几个交互式作业服务，效率高；

·格式完全自由，提供了比其他对象更好的适应性；

·既可以被 AS/400 的接口函数访问，也可以被 AS/400 的 CL 命令访问，是开发客户机/服务器程序最直接的方式。

二、数据队列技术原理

用数据队列技术实现客户机/服务器编程，可以说是通过两者的“合同”达到的。“合同”实际上就是客户机与服务器程序之间进行数据交换所作的约定。这合同在程序中指定并实现，主要有以下”条款”：

·将数据从客户机传至服务器的有效的数据队列名字；

·将数据从服务器传至客户机的有效的数据队列名字，可以与上一数据队列是 同一条队列·双方提供的控制信息在数据包中的位置；

·双方提供的数据元素在数据包中的位置及其意义；

·哪方负责提供 ASCII/EBCDIC 的转换及某些数字型变量的转化；

·传送数据时若另一方终止，该做怎样的处理。

1. 数据队列的选择

采用数据队列技术，在编程前都会碰到应该用几条数据队列的问题：一条还是两条？要回答这个问题，就必须首先弄明白两种数据队列的区别：有键数据队列和无键数据队列。

有键数据队列特点：必须使用特定的发送函数和接收函数；建立时必须指出键的长度（1－－256 字节）；数据队列中的键值不是数据包的一部分，而使用独立于数据包的单独的字段来存放。数据队列没有提供任何的机制检查输入的键值是否重复。

无键数据队列特点：建立无键数据队列时只需要确定数据的提取方式，是先进先出（FIFO）还是后进先出（LIFO）即可。

有键数据队列的键值确定多个程序中，需要接收队列中数据的那个程序。当 PC 的客户机程序把数据写入数据队列时，接口函数自动地设置服务器所需要的键值。当服务器需要从数据队列中取数时，也必须提供相应的键值。当服务器把数据放入数据队列作为对客户机的响应时，同时也包含了客户机的键值。键值通常作为客户机发给服务器的请求的一部分。

而在无键值的数据队列里，第一个程序自动取得对数据队列的操作权。实际操作时，程序几乎不可能只用一条无键数据队列来实现，这是数据队列程序的设计方法决定的：当任何一方（不论是客户机还是服务器）将数据写入数据队列后，会立即进入接收数据的循环。倘若没有键值决定谁该取数据，则很可能是 写入数据的一方又立即将数据取出。

因此使用无键数据队列的程序，通常都用两条数据队列：一条是由客户机发给服务器，而另一条则相反。

2. 控制信息与数据信息的构架

在决定了数据队列的结构（有键或无键，一条或多条）之后，接着要决定在客户机和服务器之间传递的数据

包的格式。数据包包含了控制信息和数据信息，其格式可以由程序的需要自由决定，只是两端的数据格式必须一致。

控制信息是指客户机和服务器程序之间所约定操作的符号表示。例如，对一个要求返回一条记录的简单请求来说，需要几个控制信息：取记录（送服务器），找到记录、找不到记录、无效需求（这三个返回客户机）。服务器要执行的工作越 多，控制信息也相应增加。

在服务器程序里通常建立了所需的数据结构，将这数据结构写入相应的数据队列即可。在 PC 上的客户机程序，以 VisualBasic 为例，可以利用结构数据类型格式化缓存，从缓存的字符串截取有用的数据。

3.ASCII/EBCDIC 的转换

由于 AS/400 上存储的数据记录通常是 EBCDIC 格式，而 PC 使用 ASCII 格式，要保持数据的准确性，程序就必须提供二者转换的手段。同时 AS/400 的某些数字字段的存放方式与 PC 也有差异，必须进行特别处理。

尽管可以用 AS/400 的接口函数和转换表，在 AS/400 上进行 ASCII/EBCDIC 的转换，但通常这项工作是在 PC 上完成的，理由主要有两点：

(1)PC 可以只对缓存中所需要的部分数据进行转换，而对余下数据不予理会，节省转换的资源和时间。

(2)通常 PC 拥有更多的资源进行转换处理。

ClientAccess/400 提供了可以让 VisualBasic 直接调用，含 ASCII/EBCDIC 转 换 的 扩展函数库 - - 数据转换函数（Data Transform API）。这些函数只在 PC 端运行而无需与 AS/400 进行数据交换，数据转换执行起来相当快。

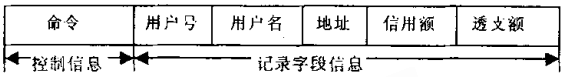
三、实例分析

理解了上述概念之后，下面来看一个以 Client Access 为接口函数，利用数据队列方式从 AS/400 读出数据到 PC 上进行操作的简单例子。

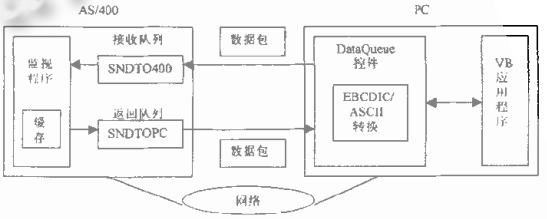
客户机程序先读出 AS/400 上存储的银行用户列表，然后由所选择的用户的编号再从 AS/400 上返回该用户的详细资料，如地址、信用额等等，并显示出来。

PC 上用 Visual Basic 编程，利用 Client Access 提供的数据队列控件（Data Queue custom control）向 AS/400 发出请求；AS/400 上则以批处理方式运行用 RPG 编写的服务器监视程序，响应请求并返回数据。

本例采用两条无键数据队列 SNDTO400 和 SNDTOPC，分别是 AS/400 接收和返回数据的队列。请求和应答均以数据包形式被传送，数据包总长 50 字节，其中最前面是控制信息，其余的是记录各字段的信息。数据包格式如下：



AS/400 上有 CUSTLNO、CUSTLNAM 两个分别以号码和名字为索引的逻辑文件，它们和监视程序及 SNDTO400、SNDTOPC 两条队列放在同一个库中。



1.服务器程序工作原理

服务器程序的主程序段是一个循环，实现对其接收队列（SNDTO400）的监视。主要的工作可以分成两个部分：接收过程和响应过程。

(1)接收过程。服务器程序开始运行后（一般以批处理方式提交，避免占用过多系统资源），进入一个“对数据队列的的接收循环”，不断地监视接收队列中是否有客户机的请求。一旦发现有客户机的请求进入数据队列，程序立即调用 OS/400 的接收函数 - - QRCVDTAQ 将请求取出，放到一个缓存中，并将其从数据队列中删除，以免重复读取。监视程序分析该请求，接着怎样执行就完全由程序根据控制信息决定。

数据队列函数对写入和读出缓存的数据基本没有限制，因此可以将任何需要的数据放入缓存。唯一的限制是缓 存最多只能存放 31,744 字节。

(2)响应过程。收到请求后，监视程序立即格式化准备回应的缓存（即填写响应数据包）。一旦缓存格式化，监视程序将调用 OS/400 的发送函数 - - QSNDDTAQ 把缓存写入相应数据队列 SNDTOPC，等待客户机取出数据。

监视程序完成将数据包写入返回数据队列后，又回

到接收循环，等待客户机的下一个请求。监视程序一般不验证客户机程序是否读取返回的数据包，对服务器监视程序来说，将返回结果写入数据队列后，就完成了使命。

响应客户机的数据队列可以就是接收请求的那条数据队列，也可以是另外指定的数据队列。服务器程序要写入客户机所需要的所有结果数据。

除了注意上述两个过程之外，也要留意在客户机程序要求退出时监视程序要清除返回数据队列(SNDTOPC)中的数据。若缺少了这个清除的步骤，客户机执行多次时将出现不可预知的结果，引起混乱。

2.客户机程序工作原理

客户机一开始就向服务器程序发出从 AS/400 取银行用户记录列表的命令，并令其按 PC 上可以处理的格式返回。当用户单击列表的其中一个记录名时，程序将列出详细资料的请求发送给服务器。

客户机程序按以下几步实现对服务器程序发出请求：

(1) 根据请求格式化一个缓存(RecBuf)作为数据包。

(2) 接着调用发送数据队列函数将数据包发送至数据队列。客户机程序要指出数据包所送至的数据队列的全称(DQSAMPLE/SNDTO400，DQSAMPLE 是 AS/400 上存放 SNDTO400 数据队列的库名)，和数据包的长度。

(3) 发送请求后，客户机程序通常进入一个接收循环。但是客户机无法知道 AS/400 的服务器程序是否真正收到了请求，只能等待响应结果放入它所监视的接收数据队列 SNDTOPC。

(4) 一旦服务器程序处理完请求并将响应结果写入返回的数据队列 SNDTOPC 后，不断循环执行的接收数据队列函数就自动地取回数据包。

(5) 客户机程序从收到的数据包中提取出所需的数据。

四、数据队列实现 C/S 编程模式的缺陷

尽管使用数据队列实现程序间的通信相当简单，但有一个难题没有解决，就是通信双方无法确认另一方是否接收到了对方发出的信息。例如，一个 AS/400 的服务器程序通常设计成对数据队列的接收循环，作为一个批处理程序运行。当数据队列接收到新的请求时，它自动对其操作，并通过数据队列返回结果给客户机，但双方程序从不直接交换信息。

对客户机程序来说，客户机程序一旦输入了请求到数据队列中后，就只能坐等服务器自己反应；即使服务器程序终止，暂停或等待系统操作信息，客户机程序对此将一无所知；当客户机程序等待服务器响应时，可能会造成死锁(由于接收函数是一种"阻塞性"函数)。

服务器程序则毫不关心客户机是否收到响应。当服务器响应客户机请求时，只是设置好数据队列的入口并写入数据，一旦数据完全写入响应队列，服务器的处理就完成了，它既不知道也不关心客户机是否取出响应数据。

针对客户机程序的死锁，有下列几种预防措施：

1. 使用非阻塞的 Put 和 ReceiveRequest 函数调用代替相应的 Send 和 Receive 函数。Put 将数据写入数据队列，但不等待 AS/400 对写入完成的确认；ReceiveRequest 能在数据队列有数据时自动取回，而不必不停地检查数据队列。程序员可以自行决定当取不到数据时所应执行的操作。

2. 对 Receive 进行等待时间的限制。当使用"永久等待"值时，程序会一直等到有新数据可取为止。假若设置了时限，则在时限到时，可让使用者决定继续等待还是放弃。

3. 程序中集成测试和恢复的函数。例如客户机可以在发送数据前执行"ping"函数，以确保与服务器的线路畅通。

要实现两端可靠的通信，也可以采用 MQSeries 等其他的中间件。

五、结束语

通过上述介绍，可以看到在客户机/服务器编程模式下，是怎样利用数据队列简单地实现 AS/400 和 PC 程序间高速数据交换的。采用数据队列技术，要求程序员熟练掌握 AS/400 上的 RPG 语言和 PC 上的高级语言，这样才能充分发挥两者的优势，真正达到高效的目的。

参考文献

[1] Darwin Boyle, Ross Rothmeier, Brad Shannon, Ron West, et al. Visual Basic 4. SAMS, 1996.

[2] IBM. AS/400 Red Book.

(来稿时间：1998 年 7 月)