银行业务系统通信模块的设计与实现*)

杨 宏[1] 蒋祥军[2] 高福祥[2]

(天津理工大学计算机科学与技术学院 天津 300191)[1](东北大学计算机信息科学与工程学院 沈阳 110004)[2]

摘 要 本文提出了银行业务系统通信的通用解决方案,该方案通过在通信模块中引入中间件技术,提高通信模块的通用性、移植性,屏蔽复杂的网络编程细节,为编程人员提供简单的编程接口,从而降低编程人员的工作量和技术难度。

关键词 通信中间件,预先派生子进程,并发

1 引言

近年来,随着银行业务的扩展,越来越多的中间业务子系统被集成到银行中心系统中。如果为每个中间业务子系统开发一个通信模块,将会导致代码冗余、结构混乱、开发效率低下、维护困难、系统可扩展性差。因此,开发一个通用的通信模块就显得特别必要。本文通过在通信模块引入中间件技术,提出银行业务系统通信的通用解决方案,并设计实现了银行业务系统的通信模块。该模块不但具有高通用性和高移植性等特点,同时还屏蔽了复杂的网络编程细节,为编程人员提供简单的编程接口,大大降低编程人员的工作量和技术难度。

2 结构模型设计

本银行综合业务系统根据 X/OPEN DTP 模型设计开发,该通信模块相当于 X/OPEN DTP 模型的通信资源管理模块,采用集中式的客户机/服务器设计模式。银行客户信息等重要数据都存储在服务器端,在客户端本地只存储像操作员信息等一些安全要求不高的数据以及一些临时数据[1]。客户端主要是负责处理与界面及输入、输出有关的工作;服务端主要是对业务的处理流程、数据库的操作进行控制。客户端根据操作员的工作,发出某项业务的请求,该请求以一个消息的形式表示,该消息通过通信中间件传送到服务端的应用程序,应用程序根据请求,执行相应的操作,然后将结果亦以消息的形式通过通信中间件传回客户端[2~4]。

在本银行综合业务系统中,将通信中间件分为两部分:服务器端通信中间件和客户端通信中间件。服务器端命名为 rsssvr,客户端命名为 rsscli,基于本通信中间件的集中式 C/S 结构如图 1 所示。

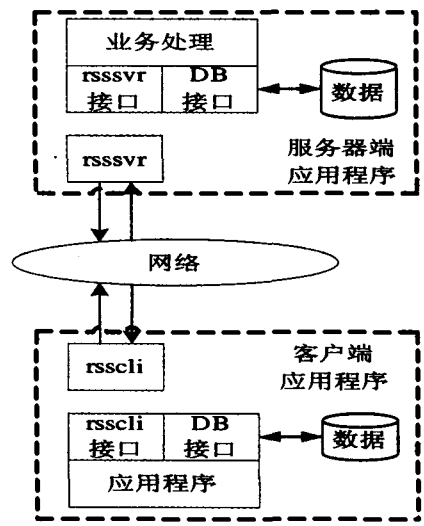

图 1 基于通信中间件的集中式 C/S 结构图

根据整体结构要求,客户端 rsscli 与服务器 rssvr 之间的通信模块采用分层设计的方案,整个通信模块分成三个层次:核心层、加密/解密层和数据报文转换层。核心层分负责对核心层状态进行监控、管理核心层资源分配以及进行数据报文的转发/接收[5]。加密/解密层对核心层传过来的数据报文进行解密操作,然后将解密后的数据报文传给数据报文转换层;而对从数据报文转换层传过来的数据报文进行加密处理,然后将加密后的数据报文传给核心层[6]。数据报文转换层负责根据数据报文传递方向对数据报文进行转换。

3 实现方案

本通信模块依据上面的分层结构划分为三个模块:数据报文转换模块、数据加密/解密模块和通信核心模块。

3.1 数据报文转换模块设计

*)基金项目:国家 863 计划(708-4-5)和辽宁省自然科学基金(项目编号 20042012)。杨 宏 讲师,硕士,从事网络服务质量控制、路由算法的研究。

在业务系统中，数据是以键-值(key-value)对的哈希表的结构存储的，当应用程序将数据传到通信层时，数据是一个哈希表。而在通信模块间，数据的传输是以数据报文的形式传输，数据报文中数据部分是一个以'|'符分隔的数据串，每两个'|'中是一个键-值(key-value)对。因此在业务层数据传到通信层时，首先要对数据进行序列化，即将以哈希表形式存放的数据转换成以'|'符分隔的字符串。然后再将序列化后的数据串传到封装数据报文模块封装成数据报文。而对于从加密/解密层传过来的数据报文，首先对数据报文进行拆解，提取出数据部分，然后再进行反序列化，将数据转换成业务系统能够识别的哈希表结构。因此数据报文转换模块又分为序列化子模块、反序列化子模块、封装数据报文子模块和拆解数据报文子模块。

(1)序列化子模块负责对从业务系统传过来的数据进行一个初始化操作，让其转换成满足封装数据报文条件的数据。其过程是指将业务系统传过来的数据转化成一串以'|'符分隔的字符串，转换后的字符串格式为：数据类型〈键，值〉|数据类型〈键，值〉|…|数据类型〈键，值〉|。转换成字符串后便于传输。

(2)反序列化是序列化过程一个逆过程，它是对拆解数据报文后的信息提部分进行提取操作，提取出信息体中以'|'分隔的各个键-值对，添加到哈希表中。最后返回此哈希表。

(3)封装数据报文子模块负责将每个待发送的数据都在封装数据报文模块将数据封装成统一的数据报文形式。根据数据报文类型，给数据添加数据报文头部和校验部分，然后传给数据加密/解密模块。

(4)拆解数据报文模块是封装数据报文模块的一个逆向过程。其负责从解密后的数据报文中提取出一次交易所有数据报文的交易码和信息体，并将其返回给调用者。对接受到的数据解密后，首先对数据进行校验，若数据正确，则取出数据报文中的数据部分。该模块默认设置要接收的数据报文状态位count为1；若他接收到的数据报文状态位值等于count，说明还有后续报文，且后续报文状态位等于count+1，则取出接收到的数据报文的信息体追加到信息体中，继续接收数据报文；若接收到的数据报文状态位等于0，说明是结束报文，在取出数据报文中的数据追加到信息体中后，将信息体返回。若接收到的数据报文状态位既不等于要接收的数据报文状态位，又不等于0，则认为是重复报文或错误报文，将丢弃该报文。

3.2 数据加密/解密模块设计

数据加密/解密模块是系统数据安全的一道重要屏障。本系统采用DES(Data Encryption Standard)算法对数据进行加密/解密。密钥为64位，这足以满足系统安全需求。此模块分为数据加密子模块、数据解密子模块和密钥管理子模块。

DES加密算法的入口参数有三个：Key，Data，Mode。其中Key为8个字节64位，有效位为56位，是DES加密算法的工作密钥；Data也为8个字节64位，是要被加密或解密的数据；Mode为DES的工作方式，有两种：加密或解密。返回结果为加密后的数据。为了便于系统使用，在本系统中编写了DES加密算法的包裹函数。此包裹函数也有三个入口参数：Key，Data，Mode。其中Key和Mode跟DES加密算法的入口参数相同，Data不再是8个字节的数据，而是任意长度的字符串。通过此包裹函数，系统可以对任意长度的字符串加密/解密。

要保证数据的安全性除了采用的加密算法稳定可靠外，密钥的管理也非常重要。本系统中采用三级密钥加密系统：传输密钥、交换密钥、主密钥。初始密钥采用随机数产生。通过给定相对安全的随机数种子，产生随机密钥。传输密钥和交换密钥由主密钥加密后存储在密钥管理数据库中，而主密钥采用移位变换后以乱码的形式存储在数据库中，因其并不用于传输数据，其安全性相对较高。为了防止密钥被盗，保护交易数据，密钥需经常更新，本系统默认更新密钥更新间隔为10天。传输密钥和主密钥在更新后需通知各个前置机更新密钥，而在传输过程中密钥可能存在风险，因此在传输密钥或主密钥在网络上传输前先由交换密钥加密，而后传到各个前置机处。而当交换密钥需更新时，由旧的交换密钥对新的交换密钥加密后传输的各个前置机处。对于通信层中的加密系统来说，不存在读取以前传输过的数据报文的情况，因此，密钥不需要备份。只需彻底将其销毁就行。

3.3 通信核心模块设计

通信核心模块是整个通信模块的重点，其性能好坏直接关系到整个通信模块的性能。通信核心模块分为管理子模块和执行子模块。采用预先派生子进程，父进程统一accept的并发服务模式。核心模块结构如图2所示。

通信管理进程(父进程)负责子进程管理、子进程调度、TCP监听管理、TCP连接管理等，是通信模块的控制中心。

子进程负责读入数据、送出数据等，同时子进程中也有简单的超时控制机制。若发送超时，则重发数据，同时记录超时次数，超过一定次数后向管理进程报告。它与父进程之间有流管道连接，用来传递

文件描述符，这是父子之间交互信息的唯一方式。

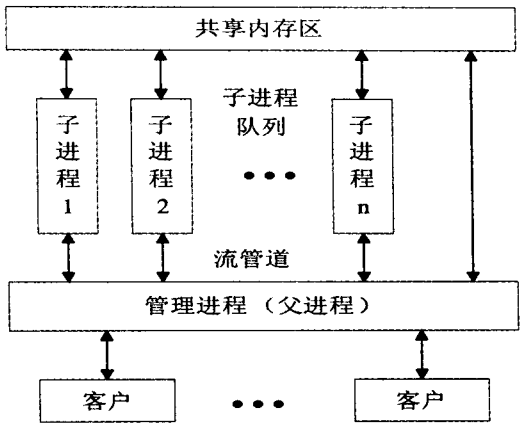

图2 核心模块结构图

共享内存区上记录着系统运行所需要的所有信息数据，是系统运行的基础。数据分为静态配置参数和动态数据。静态配置参数在系统运行期间不改变。系统在启动的时候，通过系统初始化和获取配置系统环境变量等方式获取通信系统运行相关的配置变量信息，如预启子进程的个数、在监听描述字上排队的最大客户连接数等信息，并将其全部放入核心共享内存。在系统后面的运行过程中，将只需访问共享内存，这样可以大大地提高运行效率。动态数据是类似子进程状态，各连接状态等动态变化的信息，这些数据随着系统的运行状况时刻变化。动态数据是主进程用来控制系统运行的基础信息。

客户是对服务器端提出连接请求，若连接建立，则从服务器端获得服务，最后得到响应结果的程序实体。

4 系统测试与集成

本系统分为服务器端和客户端。服务器端测试程序调用通信模块预启5个子进程，而后创建监听套接口描述字，监听客户端请求。若有客户请求，则建立连接，并将此连接套接口描述字传递给一个空闲的子进程处理，子进程只是简单地将客户端发送过来的数据回射回去，然后关闭连接，向父进程发送一个服务完成。客户端测试程序调用通信模块与服务器端建立连接，向服务器端发送200字节的数据，在收到服务器端的回应后，显示服务器端的回应数据，然后关闭连接，如此循环向服务器端请求，直到达到设定的交易数目为止。基于上述测试方案，主要针对系统的并发效率进行了测试。

首先启动服务器端应用程序，然后启动多个客户端应用程序，向服务器端进行数据请求，对服务器端形成压力测试。增加客户端应用程序数量，在服务器端查看通信进程数可知通信执行进程数不断增加。反之，减少客户端应用程序数量，通信执行进程数则减少。由此可看出通信系统能够根据客户端数量自动调节通信执行进程数量。并发测试结果如表1所示。

表1 并发效率测试结果

客户进程数	每个客户进程交易笔数	服务器端每秒交易笔数	服务子进程数
10	5000	110	2
100	500	89	5
200	250	80	5
300	150	75	8

由表1可知，在客户进程为10个时，服务器端预启子进程数自动调节为2个，而当客户进程数达到300个时服务器端管理进程预启了8个子进程，根据上述测试结果显示，能够满足中小银行需求。

结束语 本文基于银行系统业务的需要，实现了银行业务通用通信模块。该模块作为一个通信中间件设计开发，使通信系统可脱离业务系统而独立运行，系统对外提供丰富的编程接口，同时管理进程以守护进程方式运行于后台的管理进程对整个通信核心进行监控；本系统在设计过程中还采用分层的设计思想给系统实现带来便利，同时为业务系统的编程人员提供清晰明了的调用流程；系统安全部分采用DES加密算法对数据报文进行加密/解密。本通信模块只支持IPv4，对于IPv6的支持还有待于进一步研究。

参考文献

1 王加阳，王静. 基于多线程技术的银行中间业务系统[J]. 计算机应用，2002，22(10)：70～72

2 张云勇，张智江，刘锦德，等. 中间件技术原理与应用[M]. 北京：清华大学出版社，2004. 100～116

3 王加阳，王静. 基于多线程技术的银行中间业务系统[J]. 计算机应用，2002，22(10)：70～72

4 周泽华，黄涛，李京. 消息中间件管理器的设计与实现[J]. 计算机研究与发展，2002，39(3)：318～323

5 陈和平，严宇峰，方红萍. 基于DTP模型的消息中间件的设计与实现[J]. 计算机系统应用，2003(1)：40～42

6 刘晓星，胡畅霞，刘明生. 安全加密算法DES的分析与改进[J]. 微计算机信息，2006，22(12)：32～33