主编 张人凤 金竞秋 孟 临

qiusuo

求索

——上海市区办高校中青年教师论文集之四

上海科技教育出版社

求　索

——上海市区办高校中青年教师论文集之四

主　编　张人凤
　　金竟秋
　　孟　临

上海科技教育出版社

图书在版编目（CIP）数据

求索. 4：上海市区办高校中青年教师论文集 / 张人凤，金竞秋，孟临主编. —上海：上海科技教育出版社，2006.9
ISBN 7-5428-4271-4

Ⅰ. 求… Ⅱ. ①张… ②金… ③孟… Ⅲ. 高等学校—教学研究—文集 Ⅳ. G642.0-53

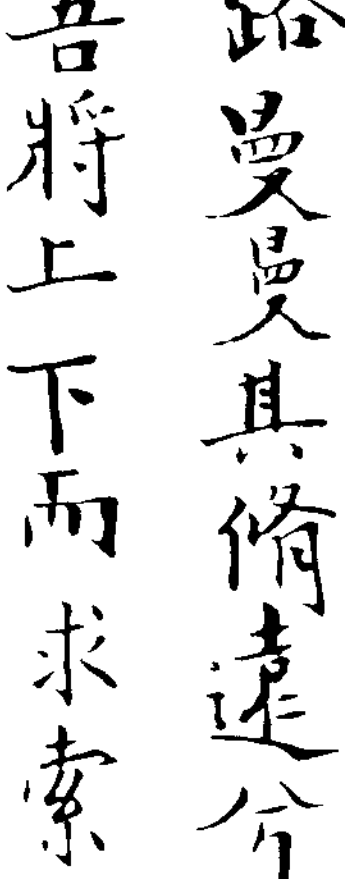
中国版本图书馆 CIP 数据核字(2006)第 102045 号

求　索

——上海市区办高校中青年教师论文集之四

主　　编：张人凤　金竞秋　孟　临
责任编辑：赵忠卫
版式设计：丁国朝
封面设计：杨颖皓

出版发行：上海世纪出版股份有限公司
上海科技教育出版社
（上海市冠生园路 393 号　邮政编码 200235）
网　　址：www.ewen.cc
www.sste.com
经　　销：各地新华书店
印　　刷：常熟文化印刷有限公司
开　　本：787×1092　1/16
字　　数：330 000
印　　张：13.25
版　　次：2006 年 9 月第 1 版
印　　次：2006 年 9 月第 1 次印刷
印　　数：1—1 000
书　　号：ISBN 7-5428-4271-4/G·2487
定　　价：50.00 元



毛泽东青年时代手书屈原《离骚》诗句

编委会名单

目录

论工作流系统中消息变量问题的处理

肖致庆

内容摘要：工作流系统是基于形式语言的人工智能的计算机模型系统。它通过将企业的业务流程用计算机可以处理的信息形式表示，来达到业务自动化的目的。一般来说，工作流系统总是基于网络的分布式结构，它由几个常用的部分组成，其中包括引擎、路由、客户程序等。

本文所涉及的工作流系统是一典型的分布程序，采用 Java 2 作为开发平台，使用 Borland Jbuilder 作为开发工具。工作流引擎是工作流系统的驱动核心，它将计算机形式的信息（KQML Message）通过网络进行传送，而客户程序必须接受引擎传送的信息进行相应的解释，呈现给用户，并且将用户的操作结果以一定的格式回送至引擎处。客户程序可以大致地分为三大部分：收发消息、处理消息和显示消息控件。而其中，比较复杂的就是工作流消息中相应变量的处理。本文通过对变量的处理进行深入的分析，解决这一复杂的问题。

关 键 词：工作流　路由　客户程序

一、课题来源及意义

工作流系统管理技术由于受到各方面的产业的开发，已经成为一项高速发展的技术。它的主要特色是流程自动化，包括人和机器活动的组合，特别是那些包括与 IT 应用程序和工具进行交互的活动。以前，工作流用于需要大量人力的办公室环境，例如：保险、银行、行政等。但是，系统现在对于一些生产和制造业也非常适用。

一个优秀的工作流系统，可以加快公司内部的业务流程，提高工作的效益，降低公司运作成本。公司业务负责人通过运用系统的功能可以花费极少的时间和精力，完成以前可能要花

费更大代价而效果又不理想的工作分配。指导者可以不必关心地点和时间，按照一定的方针分配工作，进一步保证了公司业务的顺利进行。

二、国内外研究现状

工作流管理的概念早在很多年以前就已提出，但是由于受当时计算机技术发展的限制，一直未能得到充分发展。直到最近几年，随着计算机技术和网络技术的迅猛发展，以及伴随着市场竞争的加剧，企业对提高生产质量、缩短生产周期等的强烈要求，使得工作流管理成为企业界和研究领域的热门话题，而 BPR 和 CPI 更是促进了工作流的发展。在世界各地，特别是在欧洲和北美，涌现出许多有关工作流研究的公司和组织，而功能各异的工作流管理产品更是多以百计。工作流管理不仅可以广泛应用在银行、保险、法律以及行政机关等办公自动化领域，而且也同样适用于软件开发过程的管理以及工业界和制造领域，种种迹象表明工作流管理将会对下一代的信息系统产生重大影响。

以下是一些商品化的工作流产品：

(1) ActionWorkflow System：Action Technologies 公司。有基于 Microsoft SQL Server 和 Lotus Notes 的两个版本。

(2) FlowMark：IBM 公司。可运行在 OS/2、Windows 和 AIX 上，基于 ObjectStore 数据库。

(3) WorkFlo Business System：FileNet 公司。可运行在 SunOS、UNIX、AIX、HP－UX、Macintosh和 OS/2 上，基于 Oracle 数据库。

(4) InConcert：Xerox 公司。可运行在 SunOS、AIX、DOS 和 HP－UN 上，并可使用多种数据库，如 Informix、Oracle 和 Sybase。

三、消息变量问题的处理

1. 在变量显示窗口中解决变量分离、生成问题的方法

根据客户变量列表中的变量名，搜出所有需要在显示控件里出现的工作流变量，因为客户变量的列表实际上是一个哈希表，变量的名字是其中的键，而对应的键值是“0”或“1”，前者表示变量控件不可编辑，而后者表示可以编辑。在变量中，只有文档类型不受此限制。

为了便于管理这些工作流变量，笔者定义了类 WidgeGroup，这个类实际上结构较简单，只包含两个公共变量，一个类型是 Variable，用来表示一个变量，另一个是 Jcomponent，用来表示对应变量的变量控件。因为，变量控件都是从 Jcomponent 上继承的，因此，这个类可以用来捆绑所有的工作流变量以及对应的工作流变量控件。程序中还有一个列表变量，用来连接所有的 WidgeGroup 类的实例，这个列表变量是 controlVarList，窗体的所有显示控件都包含在这个列表里，这样，程序管理这些状态各异的变量就方便多了。

在显示工作流变量之前，需要先将作为表达式的变量运算出来，运算所需的解释程序已经封装成了类，运算所需的操作数都已经蕴涵在工作项的变量列表中。因此，程序在进行显示工作之前，首先将进行运算。

显示工作流变量的第一步就是从所有的变量中分离出需要显示的变量，组成另一个队列。因为有了客户变量的列表，所以查找变量比较容易。

当变量的分离工作完成之后，就进入了生成变量的步骤。逐个查找变量，根据变量的类型

和值的个数，生成控件，具体过程如下：

当变量的类型不是文档或者布尔时，假如此变量是单值的，就生成 SingleValEdit。此变量控件最主要的就是其中的编辑框，Java 的 JtextField 控件，如果变量可编辑，就将编辑框的 Editable 设为 true，否则设定为 false，将变量的值，不管具体是何种类型，都转为字符串显示在编辑框内。用户在编辑框内对变量的值进行修改，当提交时，编辑框内的值被读出来，首先进行合法性检查。在一切正常后，将值填回变量。

当变量的类型不是文档或者布尔时，假如此变量是多值的，就生成 OptionValCombo。此变量的核心就是选择列表控件，Java 的 JcomboBox 控件，假如变量的值可以编辑，就是说，选择列表控件可以被操作，将它的 Enabled 属性设定为 true，反之，就设定为 false。如果变量可编辑，但是变量的只选性为 true，就是说选择列表控件此时只支持选择，而不支持编辑。因为，在 Java 中，选择列表不仅可以用来选择，也可以用来进行编辑。如果，变量的只选性没有规定，则选择列表控件可以进行编辑。选择控件的值来自于变量本身的信息中，如果变量中没有，则通过调用变量的数据库操作域中的内容，将所需的值从数据库中取出，添入到选择控件中。在用户完成操作后，选择控件只返回一个当前值，这个值就是变量经过选择后的值，当然，这个值也要进行合法性检查。

当变量的类型是布尔时，就不存在多值的情况。因为，只有 true，false 这两个合法的值，程序生成 SingleValCheck 控件，此控件的核心是核对框，Java 中的 JcheckBox 控件，一般情况下，控件的值是 false，即非选中状态下。此时，变量的值也只有两种，当为 true 时，选中此核对框，否则，不做任何的变化。当用户完成操作后，程序无须对值进行合法性检查，直接将 true 或 false 填回变量。

当变量的类型是文档型时，也不存在多值的情况，但是，却是处理方式最为复杂的。程序生成 DocValEdit 控件，此控件内部的子控件的数量是变化的，但大致可以归纳为：文件的固定主码，文件的描述信息，文件的非固定主码。其中，只有非固定主码是有变化的。固定主码和描述信息都用编辑框来显示，而非固定主码显示在一滚动区域中，此区域使用 JscrollPane，而主码有两种不同的显示控件，编辑框和选择框，这些控件将加入到滚动区域中去。当文档型的变量生成时，内部含有一张哈希表，此表列出此变量用到的所有的普通的变量，而且这些普通变量都是为了文档变量而存在的，他们可能是单值的，也可能是多值的，通常，程序使用编辑框和选择框来代表这些变量。至于这些变量的可否编辑性，如果变量是多值的，只允许选择，而不允许编辑，如果是单值的，假设，变量的值不为空，则不允许编辑，如果变量有值，则肯定需要编辑。其次，再根据文档的操作方式，进一步确定这些控件的可否编辑性。

如果变量的操作类型是删除、下载和更新，则所有的为变量服务的普通变量的控件都是不可编辑的，如果是选择型的，是不可以选择的。如果变量的操作类型是加入，则无须进一步改变变量控件的编辑属性。

这些为文档而生成的变量控件只有在可编辑的情况下，才需要进行合法性检查。

由于文档变量的操作存在着加入和更新，因此，变量控件中必须提供本地路径用来暂存文件。控件中含有一路径的编辑框，它的可否编辑性是根据文件的操作类型来定，只有在加入和更新，以及下载的方式下，此编辑框才可以编辑。

由于文件的具体操作是在数据库连接中定义，因此，控件编辑这一部分并不涉及文件的操

作，但文件的主码变量，因为是普通的变量，所以要进行合法性检查。

如果文档变量的操作方式是加入的话，指定本地路径后，就必须检查本地路径是否合法，并且本地路径中是否有文件。如果操作方式是下载的话，指定本地路径后，就必须将数据库中的文件下载到本地路径中去，而相应的操作方法包含在数据库连接中。如果操作方式是更新的话，首先，文件必须下载，因此，也要执行下载的操作。

在合法性检查中，除了普通的变量须进行检查之外，对于加入和更新操作，在提交通过之前，就必须检查本地路径是否合法，以及路径中是否含有对应的文件，这样，就保证了数据库连接操作时，可以正常进行。

在所有的变量控件生成完成之后，就得到了这些控件的具体尺寸，通过控件内部的函数，这些函数实现的功能就是返回本身的尺寸。用一个 Jpanel 对象作为控件的容器，将控件加入到这个 panel 中，同时，须根据当前的控件数，同步地改变 panel 的大小。当完成所有控件的添加后，将这一 panel 加入到滚动区域中去。最后，将滚动区域加在变量的显示窗体中。这样，就基本完成了变量的显示工作。

2. 变量的合法性检查

变量可以有值，也可以为空，但是布尔型的变量一定会有值。如果变量有值，就必须根据变量的类型来判定值的合法性。如果是单值且不是布尔型的，取出变量的值，此时，变量的值都是字符串形式，本人利用 Java 中的异常机制来判定值的合法性。在 Java 中，由于变量都可以生成类的实例，而各种值都可以用字符串的形式作为类构造函数的参数，如果在类生成对象的过程中发生不合法的情况，就会产生异常，程序只要获取到异常，就说明变量的值有问题，而如果类生成的过程中，一切正常，就说明以字符串为形式的变量的值是正确的。程序首先判定变量类中变量的类型，再根据类型用不同的类生成对应的对象。

如果变量是单值且是布尔类型，程序只取出核对框的当前状态，根据状态的值，就可以知道此时核对框是否选中。如果核对框已经被选中，就使用字符串"true"来作为变量的值，这是根据消息中的常规约定来设置的。如果核对框未被选中，使用字符串"false"来作为变量的值。

如果变量是多值的情况，则程序使用选择列表控件的接口取出当前选择的内容，由于所返回的值都是字符串格式，因此，接下来的处理过程与前述的相同，根据变量的类型生成不同的基本类，程序根据异常的发生情况决定值的合法性。

在进行合法性检查的时候，文档类型的变量只需处理内部的主码变量，由于这些变量的情况与以上的相同，所以，程序无须单独处理。而就文档本身来说，只要在加入和更新操作的方式下根据控件中的本地路径核对文件是否存在。

3. 提交工作流变量时程序应进行的操作

首先进行合法性检查，如果合法性检查通过，则执行下一步操作，如果合法性检查不通过，则提示相应的出错信息。

当完成合法性检查之后，程序又将进行一次变量的计算。因为如果变量是基于计算的，它所包含的计算操作数来源于当前的工作项的变量列表，尽管在显示变量控件之前，凡是需要计算的变量都已经完成计算，但假如是操作数的变量会显示出来，而且它的控件可以被编辑，用户如果修改了此变量的值，而相应的那个计算变量就必须相应地改变，而程序现在还不支持动态地改变计算变量的值，因此，在向变量列表输入最新的值之前，就必须重新计算。而显示变量之前的计算，主要是为了初始化变量，因为有些计算变量在起初是为空的，只有经过计算才

有值，为了显示的需要以及不引起误解，显示变量控件之前必须进行一次计算。

当计算完所有变量之后，此时就可以返回变量的值，应该说此时变量的值应是最新的，而且是正确的。

接下来便是完成数据库的连接操作。什么是数据库的连接操作呢？由于文档变量的存在，最终的操作只能通过数据库进行，除此之外，一些普通的工作流变量不仅通过消息返回给引擎，而且还有可能送回到数据库中，这些操作只能通过数据库连接进行。KQML 消息中负责数据库连接的消息是 DBLink，而其中真正操作是由 SQLOperation 负责。根据 SQLOperation 的消息串生成 SQLOperation 这个类，DBLink 中就会调用内部的操作类完成任务。

如果在数据库连接操作的过程中失败了，程序就不会将最后的消息结果送回引擎。当完成数据库连接操作后，程序将组织回送给引擎的消息串。根据和引擎模块之间的协调，客户端实际上只需回传变量中显示的部分。对于文档类型的变量，根据文档操作之后，文档的状态的改变与否来决定文档变量的信息最后是否回传到引擎。程序根据文档变量的操作方式来决定文档的状态是否改变。如果文件是删除操作，文档变量的状态为从无到有；如果文件是加入操作，文档变量的状态则为从有到无；如果是更新和下载操作，则文档变量的状态不变，就不需要回传至引擎。根据约定，所有为了文档而设立的普通变量都将返回到引擎端。在用户成功提交消息之后，程序将改变当前工作项的状态，从正被处理转变为已经结束。这以后，出于安全的考虑，已经完成的工作项将不允许用户再打开并且编辑。变量显示窗口将关闭，而相应的菜单条将从菜单列表中去除。同时，工作项状态统计线程将不停地搜索工作项列表，改变工作项的状态显示。

四、结论及展望

当前的工作流管理系统可以支持典型组织机构中大多数的处理过程，大约有 70%～80%。当然，还存在一些不足：许多工作流管理系统不支持异构、自治和分布环境中应用系统的集成和互操作，而一个好的工作流系统应该能够提供一种方法集成以前的应用系统，以保护过去的投资，能够灵活地支持组织机构的改组，并支持当今有关动态企业(Dynamic Enterprise)的技术；另外，在有错误产生时，工作流管理系统不能保证工作流执行的正确性和可靠性。

在过去，把数据库技术用于支持处理过程管理，例如使用触发器和存储过程，可惜的是这种工作流环境是均匀的。实际上，当前越来越迫切需要把工作流管理系统构筑在分布的、基于对象的支撑结构上，以支持大规模的企业应用，幸运的是目前有许多中间件和技术标准用于支持分布式对象计算，诸如 CORBA、DCE、DCOM、Web 和 Java RMI 等。

随着互联网的普及，Web 无处不在，价格低廉并且容易使用，因此也出现了一些基于 Web 的工作流解决方案。Web 浏览器提供了统一的、使用自然方便的用户界面，并让用户可以在任何计算平台上参与到工作流中，而不需要添加任何额外的硬件。通过对当前商品化的基于 Web 技术的工作流管理系统的观察，显示出大多数产品都仅仅是可以部分地使用 Web，但是发展的趋势是越来越多地使用面向 Web，这种趋势可以在目前的一些研究项目中体现出来，如 Action Tech Metro、WebFlo 和 DartFlow。但是由于 Web 及浏览器本身的限制，只能提供 Client/Server 计算模式，并且所使用的 CGI 接口只有有限的编程能力，在位置透明性、支持事

务功能、安全性、性能等方面还有待于进一步改善。

因此，未来的工作流管理系统的实施环境应该具有以下特点：支持异构、自治、分布的环境，能够集成老系统，支持分布对象计算，支持面向 Web 的应用，从而使整个工作流管理系统具有开放性和可重构性。另外，工作流研究是一种跨多学科的研究，涉及到人机交互、数据库、管理学、社会学等学科。任何缺乏多学科合作的研究都会阻碍工作流管理系统成为一个通用的系统，造成功能上的不足。

参 考 文 献

[1] (美)Laurence Vanhelsuwe 等. Java 从入门到精通. 北京：电子工业出版社，1997

[2] 史斌星，史佳 编著. Java 基础编程贯通教程. 北京：清华大学出版社，2003

作者单位：上海市杨浦区业余大学

基于 TCP/IP 协议的 C/S 架构网络通讯

朱 旻

内容摘要：基于 TCP/IP 协议的 C/S 架构是网络通讯领域近几年来最为流行的一种设计思路，从早期的 ICQ 到目前为网络用户普遍使用的 MSN、QQ，无一不是基于该思路设计而成。

采用 C/S 架构，使用从 MFC 类中的 CAsyncSocket 类的派生类实现底层通讯，底层利用灵活度和自由度更具优势的 UDP 数据报协议进行通讯，这样，便于客户端之间的直接通讯，也可以高效地传送消息。

关 键 词：网络通讯　Access 数据库　套接字 Socket　TCP/IP 协议　C/S 架构

一、基本介绍

1. 基于 TCP/IP 协议的 C/S 架构网络通讯

基于 TCP/IP 协议的 C/S 架构是网络通讯领域近几年来最为流行的一种设计思路，从早期的 ICQ 到目前为网络用户普遍使用的 MSN、QQ，无一不是基于该思路设计而成。为了能更深刻地了解这一设计思路，笔者将在下面探究基于 TCP/IP 协议的 C/S 架构网络通讯的众多设计细节，并尝试对设计中出现的一些问题提出自己的解决方案。

2. TCP/IP 协议

TCP/IP 是因特网进行网际互连的通信协议。TCP/IP 协议的核心是传输层协议（TCP、UDP）、网络层协议（IP）和物理接口层，这三层通常在操作系统的内核中实现。TCP/IP 网络环境下的应用程序设计是通过网络系统编程界面 Socket 实现的。Socket 提供应用程序与系统内核之间的网络编程接口。协议可以是可靠的也可以是不可靠的。传输控制协议 TCP 是一个使用三次握手机制、校验和、确认信息以及其他可靠数据传输技术的可靠协议。相比之

下，用户数据报协议 UDP 不能提供数据传输的保证机制，如在传输中出现数据报丢失，协议本身不能做出任何检测或提示。而正因为 UDP 协议这种只管发送不管检测的特性，UDP 协议具有比 TCP 协议更快的速度，更优秀的灵活度和自由度。

在 TCP/IP 术语中，端口类似于 IP 地址，IP 地址与主机地址是相联系的，端口和协议相联系。IP 数据报保存目的和源 IP 地址，同样传输协议也保存源和目的端口号。

3. C/S 架构

C/S 架构描述了一种网络程序运行的模型。该模型将网络应用程序分为客户和服务器两部分。客户方对服务器方发送信息请求，服务器方对其做出相应回答，提供服务。

4. 套接字(Socket)

Windows Sockets 是一个编程接口，它是在加州大学伯克利分校开发的套接字接口的基础上定义的。TCP/IP 的 Socket 提供下列三种类型的套接字：流式套接字、数据报套接字以及原始套接字。流式套接字定义了一种可靠的面向连接的服务，实现了无差错无重复的顺序数据传输；数据报套接字定义了一种无连接的服务，数据通过相互独立的报文进行传输，是无序的，并且不保证可靠；原始套接字则允许对低层协议如 IP 或 ICMP 等协议进行直接访问，主要用于对新的网络协议实现的测试等。

当建立服务器程序时，将服务器程序设计成等候客户的请求。为了接收客户请求，服务器程序必须对传输层的一个特定协议端口进行侦听。当服务器配置 Socket 接口时，它使用 bind()函数让 Socket 执行体登记一个协议端口。也就是说，程序告诉 Socket 执行体使用哪一个协议端口进行数据传送。Socket 执行体接着告诉传输层某个特定协议端口已被使用，并将其收到的所有数据传送给 Socket API。下面的语句显示了一个典型的函数调用：result—bind (socket _ handle, local _ structure, socket _ address, address _ length)。图 1 和图 2 所示的时序图详细显示了无连接协议与有连接协议的套接字调用过程。

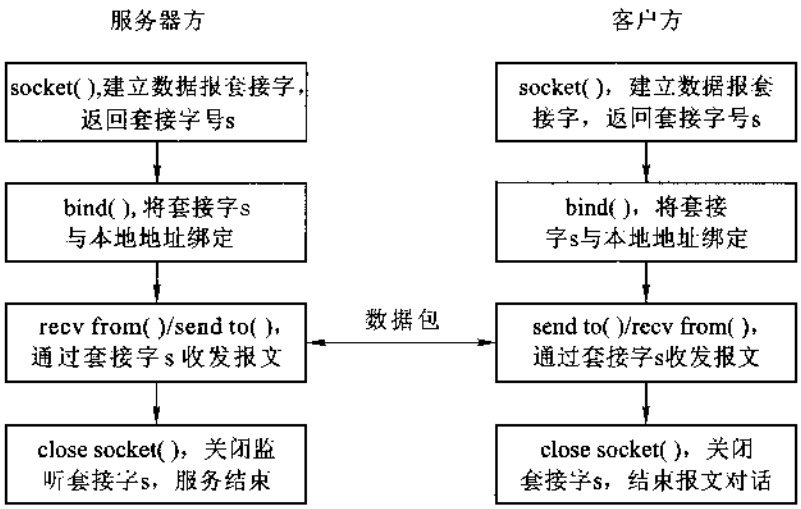


图 1 无连接协议的套接字调用时序图

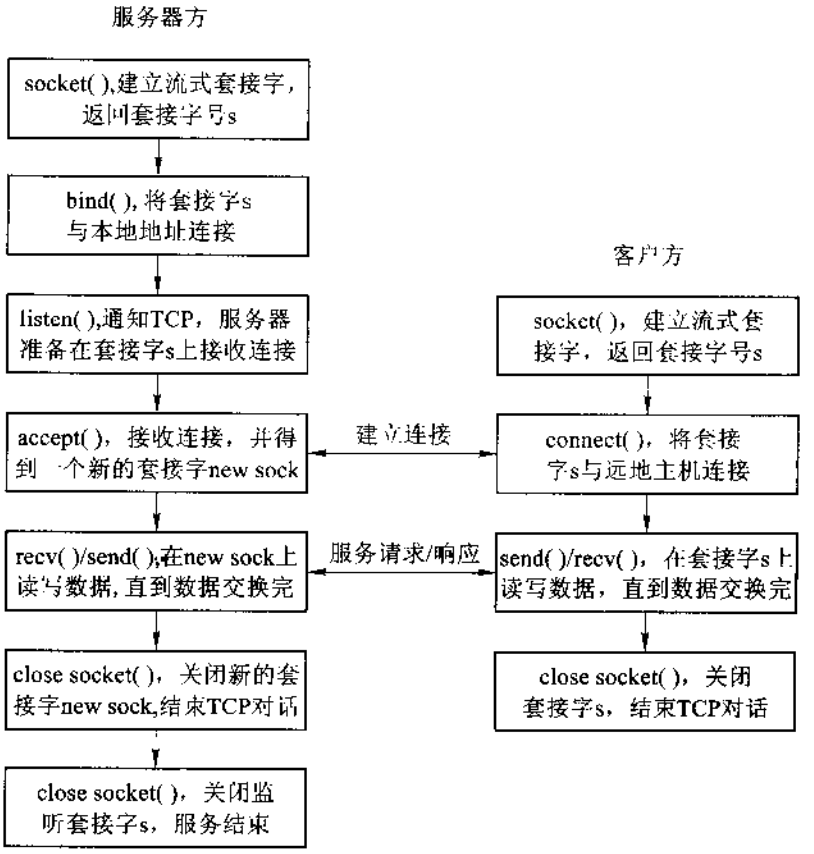


图2 面向连接的套接字系统调用时序图

二、主体开发

1. 开发工具

采用 Visual C++ 6.0 在 Windows 2000 环境下编程实现。数据库则采用较简单的 Office Access 2000。

2. 总体设计概述

采用 C/S 架构,使用从 MFC 类中的 CAsyncSocket 类的派生类实现底层通讯,底层利用灵活度和自由度更具优势的 UDP 数据报协议进行通讯,这样,便于客户端之间的直接通讯,也可以高效地传送消息。当然需注意控制其数据传输的可靠性,可以加入如下的检测方法:每发送一个数据,接收方接收到数据后,会发回一个响应信息,发送方在一个超时时间内,收到响应信息,就表示发送数据成功,若没有收到,就表示发送失败,会按用户指定的次数 N,重试 N 次,如果 N 次都失败,就返回,发送数据失败。当然,发回来的确认信息也可能丢失,但确认信息很短,相对来说,丢失的几率会小一些,是一个折中的办法。

为了保存用户信息和好友信息及一些相关数据,服务器使用到数据库技术。服务器的数据库采用的是 ODBC 的 Access 数据源,服务器访问数据库,用的是 MFC 中的 CDatabase 和 CRecordset,因为,对数据库的操作简单,使用 Sql 语句直接访问数据库,已经足够满足要求了。

服务器的运行流程为：服务器运行后，开启服务，则服务器开始侦听用户请求，如有信息发送过来，首先，发送回确认信息，然后，建立一个线程，处理接收到的数据。在线程里，按照接收到的数据类别，进行相应的处理，如有需要，会向用户发送或成功或失败的处理结果的消息，处理结束后，线程就结束了。这样，可以实时接受每个用户的请求，不会因为处理一个用户的请求，而忽略了其他用户。

客户端的运行流程为：若有本地用户信息，则取出本地用户信息，显示登录窗口；若没有，则显示用户注册窗口（在登录窗口里，也可以选择用户注册）。登录时，可选择是否隐身，进入系统后，好友列表中，在线的人将以高亮度显示，并处在列表的上头。不在线的人，将以灰色显示。登录后，如果有的话，服务器会发来好友给你发送的离线消息或广播消息。如果有好友上线了，就会通知你，好友下线了，你也可以在好友列表中看到，你可能接收到别人给你发送的消息，或广播消息等。根据用户的操作，可以向好友发送消息，查看好友信息，查看在线的人，查找用户，发送广播消息等等功能。

客户端主要是提供给用户一个友好的用户界面，方便用户操作，客户端主要负责从服务器上得到数据并显示给用户。从服务器得到好友的 IP 和 Port 后，就可以直接与好友进行通讯、聊天等等。客户端主要是界面的设计（除了底层通讯的以外），根据不同的要求，向服务器发送各种类型的请求，然后等待服务器的响应。

3. 服务器端的设计

服务器端数据库设计的要求是要能够满足客户端的需求，保存用户信息和用户好友信息、提供离线消息的服务和发广播消息的服务等。包括五个表：用户信息表（Users）、好友信息表（Friends）、广播消息表（Broadcast）、离线广播表（OffBroadcast）和离线消息表（OffMsg），因篇幅的限制，在这里就不对每个表字段的设计作详细介绍了，可根据实际情况设计满足自己要求的服务端数据库。

服务器端的底层通讯类的设计要求是能够及时响应用户的请求，当用户很多时，仍然能够适应要求，笔者把侦听与发送数据的 Socket 分开，并分别建立了多个实例，也就是说，支持多个端口的侦听，发送数据使用的是多个端口，笔者只对侦听端口感兴趣，对发送数据的端口不感兴趣，因为，发送端口是多少都无所谓。

CRecvSocket 和 CsendSocket 都是从 CAsyncSocket 类里继承而来，分别处理侦听请求各自发送数据。在 CServerSocket 类里，定义了几个 CRecvSocket 和 CSendSocket 对象的实例，通过 CServerSocket 类对内部进行组织和管理。提供给上层的接口是 CServerSocket，它隐藏了服务器底层通讯的细节及多线程发送数据的问题，提供给上一层一个统一的接口。CServerSocket 类的使用，是先建立一个它的实例，再调用成员函数 Create()，传入必要的参数。发送数据时，就调用其成员函数 SendData，处理接收数据，在 CRecvSocket 类的 OnReceive 里处理，调用了一个名为 ProcessRecvData 的线程函数，用户在这个线程函数里写上具体的处理代码。

侦听类结构：

```
class CRecvSocket : public CAsyncSocket
{
private:
  char m_szResponseMsg[MaxResponseMsgLength];      //确认消息串。
```