140多位一线编程高手智慧结晶，145个实用编程技巧典型案例解析

HZ BOOKS

电脑编程技巧与维护 杂志

2008年合订本

精华版

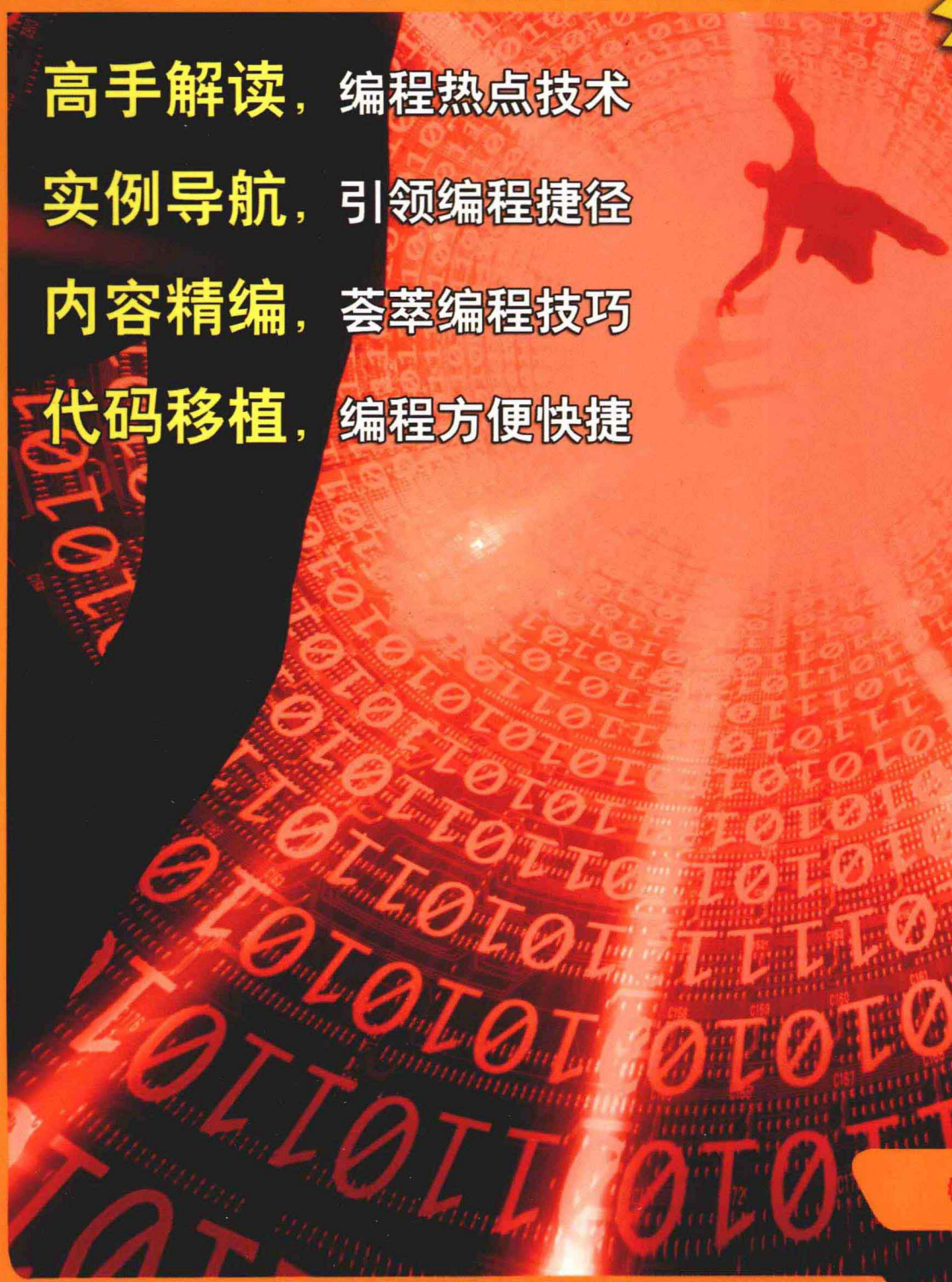
《电脑编程技巧与维护》杂志社 编

高手解读，编程热点技术

实例导航，引领编程捷径

内容精编，荟萃编程技巧

代码移植，编程方便快捷



内附光盘

机械工业出版社
China Machine Press

电脑编程技巧与维护

2008年合订本

140多位一线编程高手智慧结晶，145个实用编程技巧典型案例解析

电脑编程技巧与维护 杂志

2008年合订本

精华版

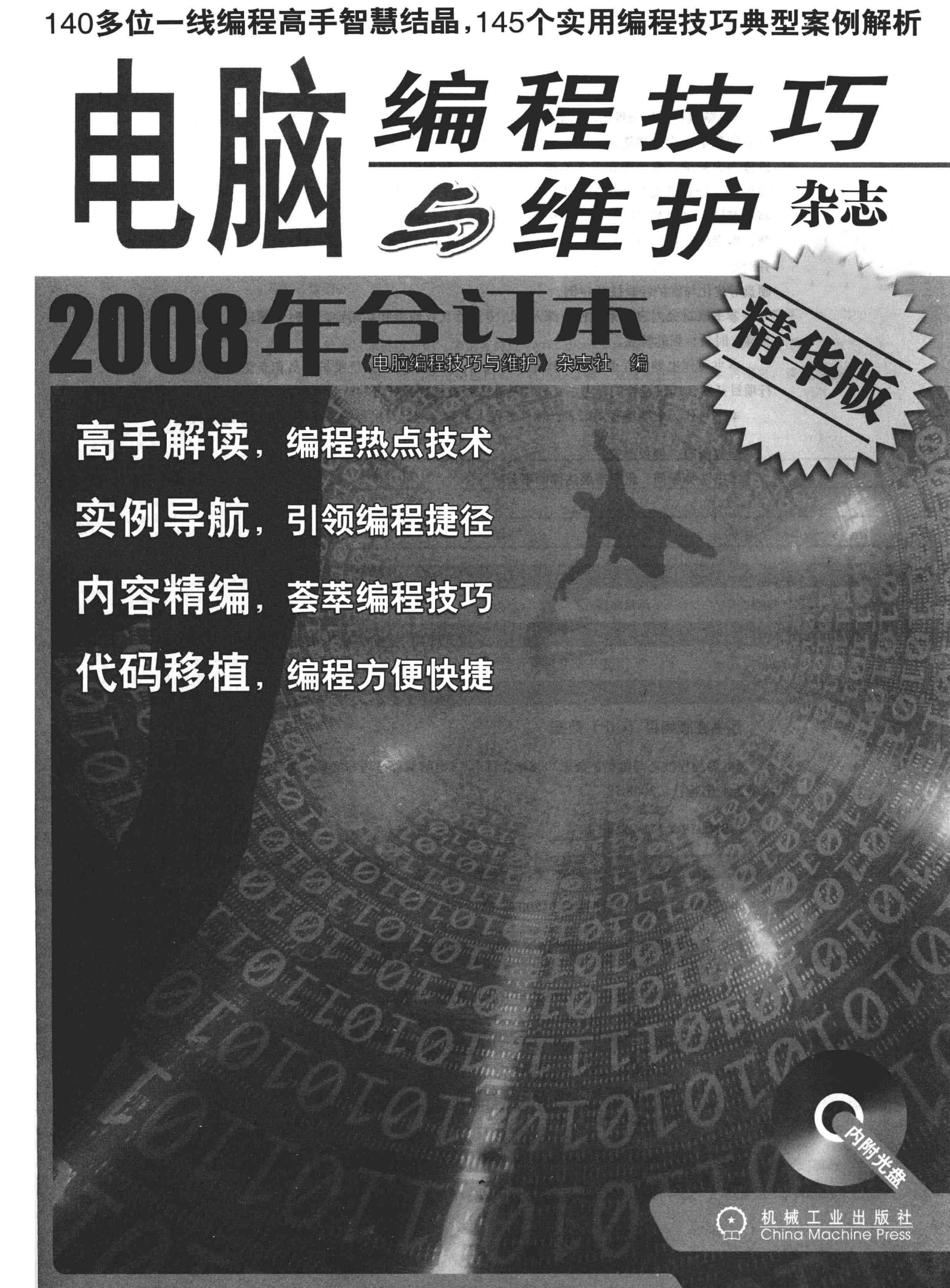
《电脑编程技巧与维护》杂志社 编

高手解读，编程热点技术

实例导航，引领编程捷径

内容精编，荟萃编程技巧

代码移植，编程方便快捷



内附光盘

机械工业出版社
China Machine Press

《电脑编程技巧与维护》杂志2008年合订本是在保留杂志原有风格的基础上，本着实用至上的原则，案例精选，清晰展示了主流编程语言的编程技术、方法与技巧：按杂志栏目设置内容精编，分为跟高手学编程、编程语言、数据库、网络与通信、图形图像处理与游戏编程、计算机安全与维护、编程疑难问题解答7篇，131个编程实例。本书以案例导航，高手解读，答疑解惑的方式，诠释编程热点技术，传授编程经验技巧，指引编程捷径。另外，本书的附录A是以Windows编程为主题，精选了14个Windows深入应用编程典型实例，并对每一个实例作了详尽地解析；附录B是电脑主要硬件设备CPU、主板、内存、显示器、硬盘等优化与维护经验技巧44例。

本书既讲究内容的系统性、深入性、专业性、权威性和实用性，同时兼顾轻松、通俗易懂、时效性强的特点。

本书可作电脑编程爱好者、软件开发人员、专业计算机系统维护人员和专业程序员进行项目开发、项目设计的参考书；软件从业人员及编程爱好者的珍藏宝典。

本书可作为电脑编程爱好者、软件开发人员和专业计算机维护人员参考书和珍藏宝典。

图书在版编目（CIP）数据

《电脑编程技巧与维护》杂志2008年合订本 /《电脑编程技巧与维护》杂志社编. —北京：机械工业出版社，2008.12

ISBN 978-7-111-25503-1

Ⅰ. 电…　Ⅱ. 电…　Ⅲ. ①程序设计　②微型计算机－维修　Ⅳ. TP311.1　TP360.7

中国版本图书馆CIP数据核字（2008）第173736号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：周茂辉

三河市明辉印装有限公司印刷·新华书店北京发行所发行

2008年12月第1版第1次印刷

210mm×285mm · 34.5印张

标准书号：ISBN 978-7-111-25503-1

ISBN 978-7-89482-874-3（光盘）

定价：69.00元（附光盘）

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

前　言

《电脑编程技巧与维护》杂志是为从事电脑编程、系统应用开发人员创办的专业性和实用性都很强的技术刊物。自1994年创刊以来，始终以“实用第一，智慧密集”为宗旨，坚持“质量第一”、“读者第一”的原则，为广大的电脑编程爱好者、软件开发人员和专业计算机系统维护人员提供第一手的技术资料、编程技巧和维护经验；紧紧跟踪计算机软硬件技术发展和应用趋势，与时俱进，不断求变创新，针对软件开发过程中许多关键技术问题，着重提供各类解决方案；在栏目内容上，选题覆盖面广，涉及技术领域宽、信息量大，帮助程序员开阔视野；在技术水平上，始终把握计算机技术发展的方向，提供先进、详尽、准确的技术指导，并在长期的工作中与国际型大公司建立了良好的合作关系，为读者提供最新的编程实用技术；在实用性上，稿源都是来自编程一线的程序员及电脑编程爱好者在项目开发实践中提炼和总结出来的经验、心得体会和技巧，是众多编程人员集体的智慧汇集。

为使《电脑编程技巧与维护》获取更大的发展空间，进一步提升杂志的品牌效益；为能够更好、更充分地利用杂志拥有的丰富而极为宝贵资源，刊登更多更好的来稿， 展现更多的应用研究成果，应广大读者的要求并考虑到杂志社可持续发展的需要，几年来，杂志社经过多方面的调研，多次的反复论证，并经国家新闻出版署审批，从2008年8月起，《电脑编程技巧与维护》变更为半月刊。上半月刊仍保持原有风格，以编程方法、技巧、经验为主要内容的电脑编程技巧典型实例解析版；下半月刊以电脑编程相关学术论文、研究成果为主要内容的学术交流版。

2008年《电脑编程技巧与维护》杂志社与机械工业出版社共同策划和倾力打造出版了《电脑编程技巧与维护》杂志2008年合订本，作为2009新年的一份礼物献给广大的读者。该合订本精华版的内容是该刊重点栏目第1期至第12期精选、精编的当前主流编程语言的典型编程案例解析；附录A是以Windows编程为主题，精选了14个Windows深入应用编程典型实例，并对每一个实例作了详尽的解析；附录B是电脑主要硬件设备CPU、主板、内存、显示器、硬盘等优化与维护经验技巧44例。本书有如下几个显著特点:

1. 高手解读，诠释热点技术

合订本汇集了140多位一线编程高手的项目开发应用经验和编程技巧，其中不少的作者是业界资深程序员和技术专家，内容有深度、思路有新意、讲解深入浅出，诠释编程热点技术，展现的编程技巧新颖实用，构思巧妙，编程技术覆盖面广，适用各类编程人员，是编程人员学习编程从有关编程书籍和网络上很难得到的学习参考资料。

2. 案例导航，突破编程瓶颈

合订本中的一个个实例都是作者从实际项目提炼出的开发范例，实例讲解部分先给出设计目标，然后介绍实现目标的基本思路和方法，最后详细给出其核心程序源代码，对其核心源代码进行解读并给出程序的运行结果。不管是初学者，还是有一定编程基础的程序员，看着实例学编程、跟着实例学技巧及解答疑惑，可较快地突破编程瓶颈。

3. 内容精编，全新编程产品

《电脑编程技巧与维护》杂志2008年合订本保留了杂志的原有风格，但不是12期内容的简单相加，是经过二次加工后形成的一个全新产品。精华合订本选取了杂志“跟高手学编程”、“编程语言”、“数据库”、“网络与通信”、“图形图像处理与游戏编程”、“计算机安全与维护”、“编程疑难问题解答”7个重点栏目中的精华文章。为方便读者阅读，其每一栏目的内容都按照编程语言的类别重新编排。

4. 代码移植，编程方便快捷

合订本中的每一个实例的程序源代码都经过上机调试通过，给程序开发人员移植源代码和学用编程带来了方便，并随书附赠一张本书所有核心源代码光盘，使读者代码移植方便快捷。

合订本既讲究内容的系统性、深入性、专业性、权威性和实用性，同时兼顾轻松、通俗易懂、时效性强的特点。品牌+品质，值得信赖；案例+技巧，助你成功。

该书可作电脑编程爱好者、软件开发人员、专业计算机系统维护人员和专业程序员进行项目开发、项目设计的参考书，也是软件从业人员及编程爱好者的珍藏宝典。

《电脑编程技巧与维护》杂志社

2008年12月

目　录

前言

第一篇　跟高手学编程

1.1 Java网络编程入门 ……1
1.2 用Java实现非阻塞的HTTP服务器 ……11
1.3 用Java实现非阻塞通信 ……20
1.4 Delphi+IntraWeb开发平台 ……30
1.5 IntraWeb 网站主页面及菜单设计……36
1.6 网站特定功能设计……41
1.7 Web数据库设计和应用 ……44
1.8 动态网站的配置与发布……48
1.9 X3D虚拟现实技术……51
1.10 X3D三维立体场景设计 ……56
1.11 X3D虚拟现实动画设计 ……64
1.12 X3D虚拟现实游戏设计 ……72

第二篇　编程语言

2.1 制作VB可读写表格自定义控件 ……79
2.2 利用VB存取数据库中BLOB数据的方法……81
2.3 Visual Basic桌面式背单词系统 ……82
2.4 用VC制作迅雷批量下载列表生成器 ……87
2.5 VC++开发邮政储蓄银行报表存储及分析系统 …91
2.6 VC++万能网考系统 ……99
2.7 VC++ 6.0自动创建树形结构 ……109
2.8 用VC++与OpenGL开发虚拟仪表控件 ……111
2.9 用Java多线程技术实现高可重用框架 ……114
2.10 Java 反编译和源代码保护……117
2.11 蚂蚁算法的Java设计与实现 ……120
2.12 C#动态区域操作界面 ……122
2.13 C#实现选择法排序的动态演示程序 ……126
2.14 Visual C#中实现约束文本编辑框……129
2.15 通过Visual C# 2005调用Cards.dll
实现扑克发牌程序……132
2.16 ASP.NET 2.0 构建课件交流平台 ……135
2.17 利用COM技术实现Delphi动态调用MATLAB…137
2.18 在.NET环境下用Treeview遍历活动目录……141
2.19 文本朗读系统……143
2.20 RFID技术在BizTalk Server中的应用 ……145
2.21 SAP中各国不同Infotype结构导出通用方案 …147
2.22 DSPI接口在BizTalk 2006上的实现 ……151

第三篇　数据库

3.1 关系数据库的VB通用查询编程技巧 ……156
3.2 流技术在VB存取工程中的应用 ……159
3.3 多视图的工程数据可视化 ……162
3.4 深入利用ASP.NET DataGrid控件 ……167
3.5 SQL Server 数据库应用程序的无值守部署 ……171
3.6 基于数据库系统的自定义文件属性 ……173
3.7 用InstallShield制作数据库软件安装包 ……177
3.8 BLOB数据类型在PB开发文件管理系统中的应用…181
3.9 网站管理分层架构技术 ……183
3.10 用SQL实现电信计费数据自动归档 ……186
3.11 数据窗口中单元格的屏蔽 ……189
3.12 利用设计模式实现数据访问的泛化……193
3.13 电子化办公工作流设计……198
3.14 PB自定义报表 ……201
3.15 准考证管理系统……205
3.16 个人资料信息管理系统……207
3.17 高考学生信息采集系统……214
3.18 Web的课题申报系统 ……219
3.19 医院电脑排队叫号系统……228

第四篇　网络与通信

4.1 程序间互联通信 ……234
4.2 测量数据的接收处理及其程序设计 ……237
4.3 MIDAS的服务器镜像技术 ……240
4.4 MFC网络蜘蛛流程分析 ……243
4.5 Java串行通信类 ……246
4.6 C/S模式中的远程方法调用……249
4.7 B/S模式中的远程调用……259
4.8 HTTP上传技术的Java实现 ……267
4.9 J2ME手机日记本的设计原理与关键技术 ……269
4.10 用C#2.0实现网络蜘蛛 ……272
4.11 使用 ASP.NET AJAX 取消服务器任务 ……279
4.12 用Delphi实现网络视频编程 ……284

4.13 提取dbx文件中的邮件信息 ……287
4.14 .NET Remoting分布式对象技术实现远程信息获取 ……290
4.15 WebService远程申报系统 ……295
4.16 USB虚拟串口通信 ……297
4.17 基于Libnids的电子邮件内容的重现 ……300
4.18 网络环境计算机上机考试管理系统 ……303

第五篇 图形图像与游戏编程

5.1 中国主干公路网最短路径查询 ……306
5.2 五子棋人工智能权重估值算法 ……310
5.3 用VC编程迷宫游戏 ……316
5.4 FreeImage的图像处理软件 ……320
5.5 用VC编程实现BMP图像裁切 ……322
5.6 色素性皮损图像的自动分割 ……324
5.7 三维游戏中相机与世界的碰撞检测及响应 ……327
5.8 Flash的UI启动画面 ……330
5.9 给DIB位图添加文本的方法 ……334
5.10 比特平面编码用于图像压缩的程序设计 ……336
5.11 双线性插值的图像缩放算法的研究与实现 ……340
5.12 实验数据图中曲线的提取 ……342
5.13 三维数字地形漫游 ……345
5.14 利用Asphyre3.1引擎实现空战小游戏 ……349
5.15 交互式、可控制图像旋转 ……352
5.16 用Delphi实现可视化答题卡设计器 ……354
5.17 用MATLAB实现数字图像水印 ……362
5.18 应用COM技术扩展ArcGIS的功能 ……365
5.19 VRML实现三维机器人仿真模拟 ……371

第六篇 计算机安全与维护

6.1 用VC制作“每日提示”对话框 ……375
6.2 U盘病毒及其免疫程序 ……377
6.3 远程开机关机 ……381
6.4 Windows服务编写综述 ……385
6.5 使用ZLIB开发的WINZIP文件管理器 ……391
6.6 主机自动备份U盘文件程序设计与实现 ……395
6.7 Visual C# 2005的自定义登录验证框 ……398
6.8 网络硬盘的C#设计与实现 ……403
6.9 角色访问动态生成用户权限菜单树 ……406
6.10 Delphi编写服务程序实现机房机器自动更新 ……410
6.11 内存映射文件的操作 ……413
6.12 USB存储设备监控程序的开发 ……417
6.13 高性能的文件加密系统 ……421
6.14 进程管理器 ……426
6.15 利用反射方法操作Windows窗体 ……436
6.16 系统登录认证技术 ……440
6.17 用Win32汇编语言对PE格式的EXE文件进行口令加密 ……446

第七篇 编程疑难问题解答

7.1 怎样使用 Visual Basic 宏处理Excel重复记录 ……454
7.2 怎样用VB实现屏幕抓图功能 ……455
7.3 如何制作试卷生成系统 ……456
7.4 怎样实现类似flashget浮动窗口功能 ……458
7.5 在Windows环境下如何单独设置多显示器的分辨率和刷新频率 ……460
7.6 如何自动更新框上的版本信息 ……461
7.7 用VC实现MSCOMM32 控件的自动注册 ……463
7.8 如何利用V C++自动生成Excel表格 ……464
7.9 如何用VC++ 6.0编程实现文件分割器 ……465
7.10 如何利用VC++实现多文件合并与任意文件的提取 ……467
7.11 怎样用VC++实现多媒体文件信息的批量录入 ……469
7.12 怎样利用WinInet技术开发FTP客户端程序 ……470
7.13 如何利用C#实现椭圆位图绘制 ……471
7.14 用delphi实现swf文件和图像合成 ……473
7.15 怎样实现基于VB的平滑滚动字幕 ……475
7.16 如何创建和访问MFC动态链接库 ……476
7.17 如何用D3D实现流体仿真 ……478
7.18 怎样在3D模型上绘制二维平面信息 ……479
7.19 如何对Excel编程实现考试成绩的统计分析 ……480
7.20 怎样在DataGrid控件中内嵌DropDownList子控件 ……481
7.21 如何用VB.NET实现点阵数据的生成与上传 ……483
7.22 如何利用图片加密文本文件 ……484
7.23 怎样用Ajax实现网站输入框的自动提示功能 ……486
7.24 怎样在VFP应用中播放背景音乐 ……488
附录A Windows深入应用编程典型实例精解14例 ……490
附录B 电脑主要硬件设备最新优化与维护经验技巧44例 ……529

第一篇　跟高手学编程

1.1 Java网络编程入门

◎ 孙卫琴

一、进程通信

进程是指运行中的程序，进程的任务就是执行程序中的代码。下面的EchoPlayer类是一个独立的Java程序，它可以在任意一台安装了JDK的主机上运行。EchoPlayer类不断读取用户从控制台输入的任意字符串XXX，然后输出echo:XXX。如果用户输入的字符串为“bye”，就结束程序。EchoPlayer类的代码如下：

```
import java.io.*;
public class EchoPlayer {
  public String echo(String msg) {
    return "echo:"+msg; }
  public void talk()throws IOException {
    BufferedReader br=new BufferedReader(new InputStreamReader
      (System.in));
    String msg=null;
    while((msg=br.readLine())!=null){
      System.out.println(echo(msg));
      if(msg.equals("bye")) //当用户输入"bye"，结束程序
        break;
    }
  }
  public static void main(String arg[])throws IOException{
    new EchoPlayer().talk();
  }
}
```

运行“java EchoPlayer”命令，就启动了EchoPlayer进程，该进程执行EchoPlayer类的main()方法。图1演示了EchoPlayer进程的运行过程，它从本地控制台中获得标准输入流和标准输出流。本地控制台为用户提供了基于命令行的用户界面，用户通过控制台与EchoPlayer进程交互。

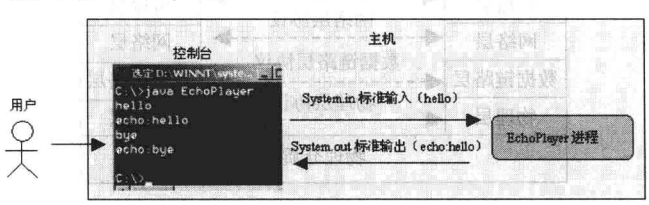


图1　EchoPlayer进程的运行过程

EchoPlayer类的echo(String msg)方法负责生成响应结果。如果需要把生成响应结果的功能（即echo(String msg)方法）移动到一个远程主机上，那么上面的EchoPlayer类无法满足这一需求。在这种情况下，要创建两个程序：客户程序EchoClient和服务器程序EchoServer。EchoClient程序有两个作用：与用户交互，从本地控制台获得标准输入流和标准输出流。与远程的EchoServer通信，向EchoServer发送用户输入的字符串，接收EchoServer返回的响应结果，再把响应结果写到标准输出流。

EchoServer程序负责接收EchoClient发送的字符串，然后把响应结果发送给EchoClient。图2演示了EchoClient与EchoServer的通信过程。客户机和远程服务器是通过网络连接的两台主机，客户机上运行EchoClient进程，远程服务器上运行EchoServer进程。

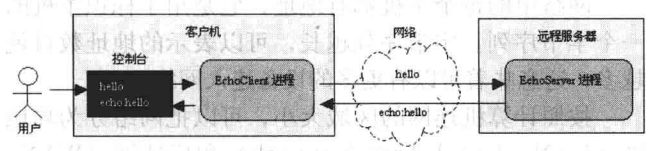


图2　EchoClient与EchoServer的通信过程

张三给李四打电话，两者顺利通话的前提条件是他们各自的电话机都连接到了电话网络上。张三和李四只需关注他们谈话的具体内容，而不必考虑如何把自己的话音传输到对方的电话机上。传输语音信息的任务是由电话网络来完成的。

同样，两个进程顺利通信的前提条件是它们所在的主机都连接到了计算机网络上。EchoClient与EchoServer只需关注它们通信的具体内容，例如EchoClient发送信息“hello”，那么EchoServer返回信息“echo:hello”。EchoClient和EchoServer都无需考虑如何把信息传输给对方。传输信息的任务是由计算机网络来完成的。

Java开发人员的任务是编写EchoClient和EchoServer程序，接下来在两台安装了JDK的主机上分别运行它们，两个进程就会有条不紊地通信。

由于进程通信建立在计算机网络的基础上，Java开发人员有必要对计算机网络有基本的了解，这有助于更容易地掌握Java网络编程技术。

二、计算机网络

计算机网络是现代通信技术与计算机技术相结合的产物。所谓计算机网络，是指把分布在不同地理区域的计算机用通信线路互联起来的一个具有强大功能的网络系统。在计算机网络上，众多计算机可以方便地互相通信、共享硬件、软件和数据信息等资源。通俗地说，计算机网络就是通过电缆、电话线、或无线通信设施等互联的计算机的集合。

网络中每台机器称为节点（node）。大多数节点是计算机，此外，打印机、路由器、网桥、网关和哑终端等也是节点。在本书中，用节点一词指代网络中的任意一个设备，用主机指代网络中的计算机节点。

如图3所示，人与人之间通过某种语言来交流，网络中的主机之间也通过“语言”来交流，这种语言称为网络协议，这是对网络协议的通俗解释，后面还会更深入地介绍网络协议的概念。

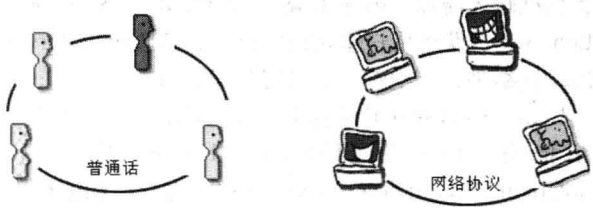


图3　网络协议是网络中主机之间通信的语言

网络中的每个主机都有地址，它是用于标识主机的一个字节序列。字节序列越长，可以表示的地址数目就越多，这意味着可以有更多的设备连入网络。

按照计算机连网的区域大小，可以把网络分为局域网（LAN，Local Area Network）和广域网（WAN，Wide Area Network）。局域网（LAN）是指在一个较小地理范围内的各种计算机互联在一起的通信网络，可以包含一个或多个子网，通常局限在几千米的范围之内。例如在一个房间、一座大楼，或是在一个校园内的网络可称为局域网。广域网（WAN）连接地理范围较大，常常是一个国家或是一个洲，其目的是为了让分布较远的各局域网互联。

到Internet海洋去冲浪，如今已成为一种时尚。Internet是指国际互联网，也叫做因特网，是全球范围内的广域网，为世界上不同类型的计算机交换数据提供了通信媒介。

Internet目前已经覆盖了160多个国家和地区，连接着几十万个子网和上千万台电脑主机，Internet的用户超过4000万。Internet上汇聚了成千上万的信息资源，成为世界上信息资源最丰富的公共计算机网络。

Internet提供的服务包括WWW（World-Wide-Web）服务、电子邮件（E-mail）服务、文件传输（FTP）服务和远程登录（Telnet）服务等，全球用户可以通过这些服务，来获取Internet上的信息或者开展各种业务。

Internet是由许多小的网络互联成的国际性大网络，在各个小网络内部使用不同的协议，那么如何使不同的网络之间能进行信息交流呢？如图4所示，上海人讲上海方言，广东人讲广东方言，上海人与广东人用普通话沟通。与此相似，不同网络之间的互联靠网络上的标准语言—TCP/IP协议。如图5所示，一个网络使用协议A，另一个网络使用协议B，这两个网络通过TCP/IP协议进行互联。

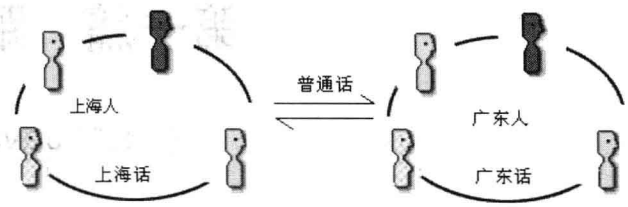


图4　上海人与广东人用普通话沟通

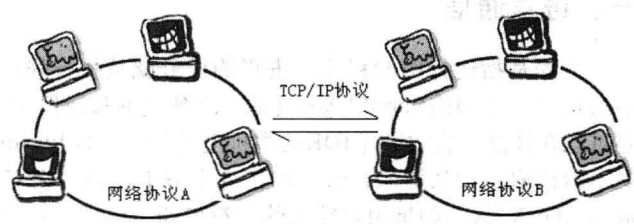


图5　不同的网络通过TCP/IP协议互联

三、OSI参考模型

在计算机网络产生之初，每个计算机厂商都有一套自己的网络体系结构，它们之间互不相容。为此，国际标准化组织（ISO，International Organization for Standization）在1979年建立了一个分委员会，来专门研究一种用于开放系统互联（Open System Interconnection，OSI）的体系结构，“开放”这个词意味着：一个网络系统只要遵循OSI模型，就可以和位于世界上任何地方的、也遵循OSI模型的其他网络系统连接。这个分委员会提出了OSI参考模型，它为各种异构系统互联提供了概念性的框架。

OSI参考模型把网络分为7层，分别是物理层、数据链路层、网络层、传输层、会话层、表示层和应用层，参见图6。每一层使用下层提供的服务，并为上层提供服务。

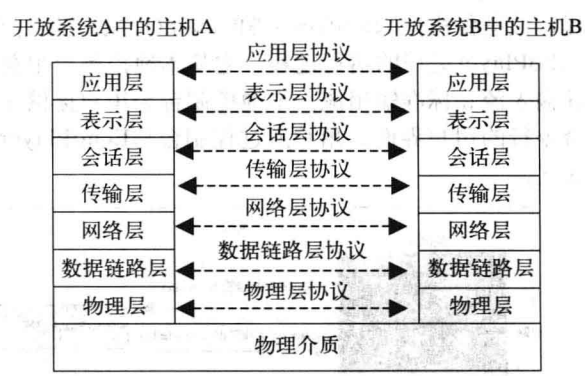


图6　OSI参考模型的分层结构

不同主机之间的相同层称为对等层。例如主机A中的表示层和主机B中的表示层互为对等层，主机A中的会话层和主机B中的会话层互为对等层。

OSI参考模型中各层的主要功能如下。

1. 物理层（Physical Layer）

传输信息离不开物理介质，如双纽线和同轴电缆等，但物理介质并不在OSI的7层之内，有人把物理介质当作OSI的第零层。物理层的任务就是为它的上一层提供物理连接，以及规定通信节点之间的机械和电气等特性，如规定电缆和接头的类型、传送信号的电压等。在这一层，数据作为原始的比特（bit）流传输。本层的典型设备是Hub（集线器）。

2. 数据链路层（Data Link Layer）

数据链路层负责在两个相邻节点间的线路上，无差错的传送以帧为单位的数据。每一帧包括一定数量的数据和一些必要的控制信息。数据链路层要负责建立、维持和释放数据链路的连接。在传送数据时，如果接收方检测到所传数据中有差错，就要通知发送方重发这一帧。本层的典型设备是Switch（交换机）。

3. 网络层（Network Layer）

在计算机网络中进行通信的两个计算机之间可能会经过很多个数据链路，也可能还要经过很多通信子网。网络层的任务就是选择合适的网间路由和交换节点，确保数据及时传送到目标主机。网络层将数据链路层提供的帧组成数据包，包中封装有网络层包头，包头中含有逻辑地址信息—源主机和目标主机的网络地址。本层的典型设备是Router（路由器）。

如图7所示，主机A发送的数据先后经过节点1和节点4，最后到达主机B。相领两个节点之间的线路称为数据链路，比如主机A与节点1、节点1与节点4，以及节点4与主机B之间的线路都是数据链路。数据链路层负责数据链路上的数据传输。从主机A到主机B的整个路径称为路由，网络层负责选择合适的路由。

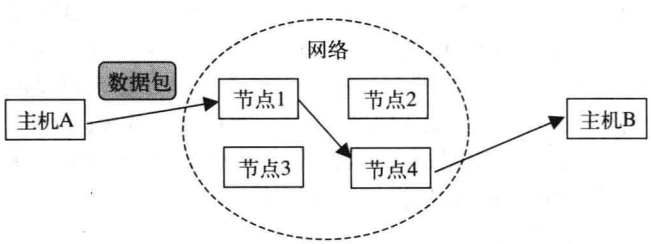


图7　从主机A到主机B的路由以及数据链路

4. 传输层（Transport Layer）

该层的任务是根据通信子网的特性最佳地利用网络资源，为两个端系统（也就是源主机和目标主机）的会话层提供建立、维护和取消传输连接的功能，以可靠方式或者不可靠方式传输数据。所谓可靠方式，是指保证把源主机发送的数据正确地送达目标主机；所谓不可靠方式，则是指不保证把源主机发送的数据正确地送达目标主机，数据有可能丢失，或出错。在这一层，信息的传送单位是报文。

5. 会话层（Session Layer）

这一层也可以称为会晤层或对话层，在会话层及以上层次中，数据传送的单位不再另外命名，统称为报文。会话层管理进程之间的会话过程，即负责建立、管理、终止进程之间的会话。会话层还通过在数据中插入校验点来实现数据的同步。

6. 表示层（Presentation Layer）

表示层对上层数据进行转换，以保证一个主机的应用层的数据可以被另一个主机的应用程理解。表示层的数据转换包括对数据的加密、解密、压缩、解压和格式转换等。

7. 应用层（Application Layer）

应用层确定进程之间通信的实际用途，以满足用户实际需求。浏览Web站点、收发E-mail、上传或下载文件以及远程登录服务器等都可以看作是进程之间通信的实际用途。

如图8所示，当源主机向目标主机发送数据，在源主机方，数据先由上层向下层传递，每一层会给上一层传递来的数据加上一个信息头（header），然后向下层发出，最后通过物理介质传输到目标主机，在目标主机方，数据再由下层向上层传递，每一层先对数据进行处理，把信息头去掉，再向上层传输，最后到达最上层，就会还原成实际的数据。各个层加入的信息头有着不同的内容，比如网络层加入的信息头中包括源地址和目标地址信息；传输层加入的信息头中包括报文类型、源端口和目标端口、序列号和应答号等。在图8中，AH、PH、SH、TH、NH和DH分别表示各个层加入的信息头，数据链路层还会为数据加上信息尾DT。

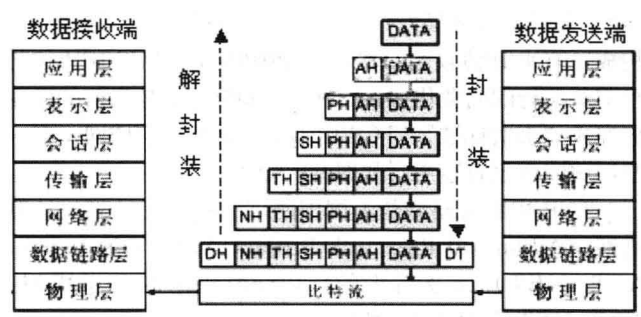


图8　数据在上下层之间的封装和解封装过程

在发送方，数据由上层向下层传递，每一层把数据封装后再传给下层；在接收方，数据由下层向上层传递，每一层把数据解封装后再传给上层。在生活中，也常常采用这种方式来传输实际物品。比如张三给李四邮寄一封信，真正要传输的内容是信，为了保证信能正确到达目的地，在发送方，需要把信封装到一个信封中，上面写上发信人和收信人地址。邮件到了接收方，需要拆开信封，才能得到里面的信件。

OSI参考模型把网络分为多个层次，每个层次有明确的分工，这简化了网络系统的设计过程。例如在设计应用层时，只需考虑如何创建满足用户实际需求的应用，在设计传输层时，只需考虑如何在两个主机之间传输数

据，在设计网络层时，只需考虑如何在网络上找到一条发送数据的路径，即路由。

对等层之间互相通信需要遵守一定的规则，如通信的内容和通信的方式，这种规则称为网络协议(Protocol)。值得注意的是，OSI参考模型并没有具体的实现方式，它没有在各层制定网络协议，但它为其他计算机厂商或组织制定网络协议提供了参考框架。网络的各个层次都有相应的协议，以下归纳了OSI各个层的一些典型协议，这些协议均由第三方提供：

- 物理层协议：EIA/TIA RS-232、EIA/TIA RS-449、V.35、RJ-45等。
- 数据链路层协议：SDLC、HDLC、PPP、STP、帧中继等。
- 网络层协议：IP、IPX、RIP、OSPF等。
- 传输层协议：TCP、UDP、SPX等。
- 会话层协议：NetBIOS、ZIP（AppleTalk区域信息协议）等。
- 表示层协议：ASCII、ASN.1、JPEG、MPEG等。
- 应用层协议：TELNET、FTP、HTTP、SNMP等。

四、TCP/IP参考模型及协议

ISO制定的OSI参考模型提出了网络分层的思想，这种思想对网络的发展具有重要的指导意义。但由于OSI参考模型过于庞大和复杂，使它难以投入到实际运用中。与OSI参考模型相似的TCP/IP参考模型吸取了网络分层的思想，但是对网络的层次做了简化，并且在网络各层（除了主机－网络层外）都提供了完善的协议，这些协议构成了TCP/IP协议集，简称TCP/IP协议。TCP/IP协议是目前最流行的商业化协议，相对于OSI，它是当前的工业标准或“事实标准”，TCP/IP协议主要用于广域网，在一些局域网中也有运用（如图9所示）。

TCP/IP参考模型是美国国防部高级研究计划局计算机网（Advanced Research Project Agency Network，ARPANET）以及后来的Internet使用的参考模型。ARPANET是由美国国防部（U.S. Department of Defense，DoD）赞助的研究网络。最初，它只连接了美国境内的四所大学。随后的几年中，它通过租用的电话线连接了数百所大学和政府部门。最终，ARPANET发展成为全球规模最大的互联网络—Internet。最初的ARPANET则于1990年永久关闭。图9把TCP/IP参考模型和OSI参考模型作了对比。

1. TCP/IP参考模型

TCP/IP参考模型分为四个层次：应用层、传输层、网络互联层和主机－网络层。在每一层都有相应的协议。确切地说，TCP/IP协议应该称为TCP/IP协议集，它是TCP/IP参考模型的除了主机－网络层以外的其他三层的协议的集合，而IP协议和TCP协议则是协议集中最核心的两个协议。表1列出了各层的主要协议，其中主机－网络层的协议是由第三方提供的。

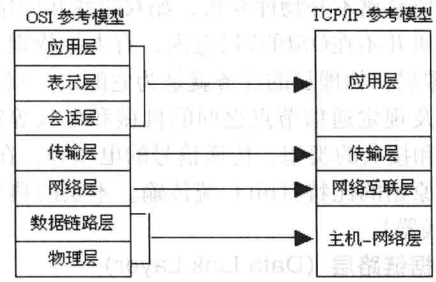


图9 比较TCP/IP参考模型和OSI参考模型

表1 TCP/IP参考模型的各层的协议

层	协议名	
应用层	FTP、TELNET、HTTP	SNMP、DNS
传输层	TCP	UDP
网络互联层	IP	
主机－网络层	以太网：IEEE802.3 令牌环网：IEEE802.4	

在TCP/IP参考模型中，去掉了OSI参考模型中的会话层和表示层，这两层的功能被合并到应用层，同时将OSI参考模型中的数据链路层和物理层合并到主机－网络层。下面分别介绍各层的主要功能。

（1）主机－网络层

实际上TCP/IP参考模型没有真正提供这一层的实现，也没有提供协议。它只是要求第三方实现的主机－网络层能够为上层—网络互联层提供一个访问接口，使得网络互联层能利用主机－网络层来传递IP数据包。

IEEE（Institute of Electrical and Electronics Engineers，美国电气及电子工程师学会）制定了IEEE802.3和IEEE802.4协议集，它们位于OSI参考模型的物理层和数据链路层，相当于位于TCP/IP参考模型的主机－网络层。采用IEEE802.3协议集的网络称为以太网，采用IEEE802.4协议集的网称为令牌环网。以太网和令牌环网都向网络互联层提供了访问接口。

（2）网络互联层

网络互联层是整个参考模型的核心。它的功能是把IP数据包发送到目标主机。为了尽快地发送数据，IP协议把原始数据分为多个数据包，然后沿不同的路径同时传递数据包。如图10所示，由主机A发出的原始数据被分为三个数据包，然后沿不同的路径到达主机B，可谓

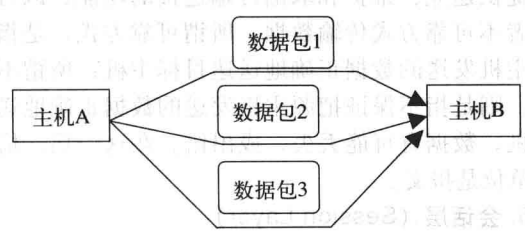


图10 三个数据包沿不同的路径到达主机B

殊途同归。数据包到达的先后顺序和发送的先后顺序可能不同，这就需要上层——传输层对数据包重新排序，还原为原始数据。

网络互联层具备连接异构网的功能。图11显示了连接以太网和令牌环网的方式。以太网和令牌环网是不同类型的网，两者有不同的网络拓扑结构。以太网和令牌环网都向网络互联层提供了统一的访问接口，访问接口向网络互联层隐藏了下层网络的差异，使得两个网络之间可以顺利传递数据包。

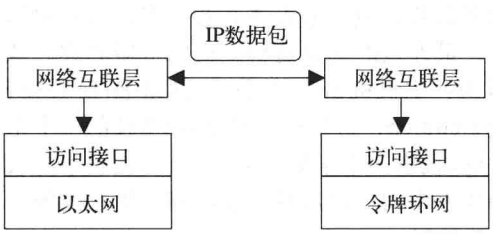


图11　网络互联层连接以太网和令牌环网

网络互联层采用IP协议（Internet Protocol），它规定了数据包的格式，并且规定了为数据包寻找路由的流程。

（3）传输层

传输层的功能是使源主机和目标主机上的进程可以进行会话。在传输层定义了两种服务质量不同的协议，即TCP（Transmission Control Protocol，传输控制协议）和UDP（User Datagram Protocol，用户数据报协议）。TCP协议是一种面向连接的、可靠的协议。它将源主机发出的字节流无差错地发送给互联网上的目标主机。在发送端，TCP协议负责把上层传送下来的数据分成报文段并传递给下层。在接收端，TCP协议负责把收到的报文进行重组后递交给上层。TCP协议还要处理端到端的流量控制，以避免接收速度缓慢的接收方没有足够的缓冲区来接收发送方发送的大量数据。应用层的许多协议，如HTTP、FTP和TELNET协议等都建立在TCP协议基础上。

UDP协议是一个不可靠的、无连接协议，主要适用于不需要对报文进行排序和流量控制的场合。UDP不能保证数据报的接收顺序同发送顺序相同，甚至不能保证它们是否全部到达目标主机。应用层的一些协议，如SNMP和DNS协议就建立在UDP协议基础上。如果要求可靠地传输数据，则应该避免使用UDP协议，而要使用TCP协议。

（4）应用层

TCP/IP模型将OSI参考模型中的会话层和表示层的功能合并到应用层实现。针对各种各样的网络应用，应用层引入了许多协议。基于TCP协议的应用层协议主要包括：

- FTP（File Transfer Protocol）：文件传输协议，允许在网络上传输文件。
- TELNET：虚拟终端协议，允许从主机A登录到远程主机B，使得主机A充当远程主机B的虚拟终端。
- HTTP（Hyper Text Transfer Protocol）：超文本传输协议，允许在网络上传送超文本。
- HTTPS（Secure Hypertext Transfer Protocol）：安全超文本传输协议，允许在网络上安全地传输超文本，网上传输的是经过加密的数据，到达目的地后再对数据解秘。
- POP3（Post Office Protocol - Version 3）：邮局协议－版本3，允许用户在客户程序中访问在远程服务器上的电子邮件。
- IMAP4（Internet Message Access Protocol Version 4）：Internet消息访问协议－版本4，允许用户访问和操纵远程服务器上的邮件和邮件夹。IMAP4改进了POP3的不足，用户可以通过浏览信件头来决定是不是要下载此信件，还可以在服务器上创建或更改文件夹或邮箱，删除信件或检索信件的特定部分。在POP3中，信件是保存在服务器上的，当用户阅读信件时，所有内容都会被立刻下载到用户的机器上。IMAP4服务器可以看成是一个远程文件服务器，而POP3服务器可以看成是一个存储转发服务器。
- SMTP（Simple Mail Transfer Protocol）：简单邮件传送协议，是发送电子邮件的协议。

基于UDP协议的应用层协议主要包括：

- SNMP（Simple Network Management Protocol）：网络管理协议，为管理本地和远程的网络设备提供了一个标准化途径，是分布式环境中的集中化管理协议。
- DNS（Domain Name System）：域名系统协议，把主机的域名转换为对应的IP地址。

2. IP协议

IP网络（即在网络层采用IP协议的网）中每台主机都有惟一的IP地址，IP地址用于标识网络中的每个主机。IP地址是一个32位的二进制序列。为了便于在上层应用中方便地表示IP地址，可以把32位的二进制序列分为四个单元，每个单元占8位，然后用十进制整数来表示每个单元，这些十进制整数的取值范围是0~255。如某一台主机的IP地址可为：192.166.3.4。

IP地址由两部分组成：IP网址和IP主机地址。IP网址表示网络的地址，IP主机地址表示网络中的主机的地址。网络掩码用来确定IP地址中哪部分是网址，哪部分是主机地址。

网络掩码的形式与IP地址相同，但有一定的限制。在网络掩码的二进制序列中，前面部分都为1，后面部分都为0。假定IP地址192.166.3.4的网络掩码为255.255.255.0。这个网络掩码的二进制序列为11111111.11111111. 11111111.00000000。把网络掩码与IP地址进行二进制与操作，得到的结果就是IP网址。因此，IP地址192.166.3.4的网址为192.166.3.0。如果把网络掩码设为255.255.0.0，那么IP网址为192.166.0.0。

图12显示了两个互联的网络的配置，从该图可以看出，每个网络都有IP网址，两个网络之间用路由器连接。

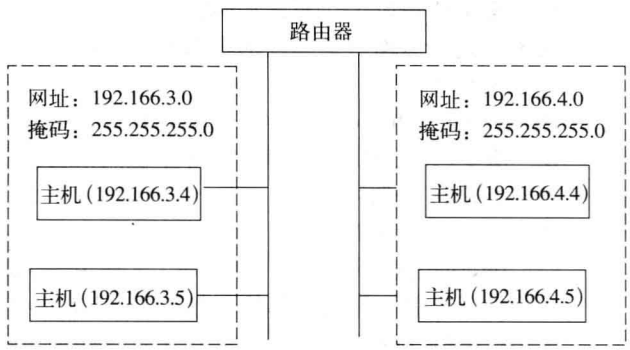


图12 每个IP网络有自己的网址，通过路由器与其他网络连接

（1）发送数据包的过程

IP是面向包的协议，即数据被分成若干小数据包，然后分别传输它们。IP网络上的主机只能直接向本地网上的其他主机（也就是具有相同IP网址的主机）发送数据包。主机实际上有两个不同性质的地址：物理地址和IP地址。物理地址是由主机上的网卡来标识的，物理地址才是主机的真实地址。如图13所示，主机A向同一个网络上的另一个主机B发包时，会通过地址解析协议（ARP，Address Resolution Protocol）获得对方的物理地址，然后把包发给对方。ARP协议的运行机制为：主机A在网络上广播一个ARP消息："要寻找地址为192.166.3.5的主机"，接着，具有这个IP地址的主机B就会做出响应，把自身的物理地址告诉主机A。

图13 在同一个网络中，主机A直接向主机B发送IP数据包

当主机A向另一个网络上的主机B发送包时，主机A利用ARP协议找到本地网络上的路由器的物理地址，把包转发给它。路由器会按照如下步骤处理数据包：

1）如果数据包的生命周期已到，则该数据包被抛弃。

2）搜索路由表，优先搜索路由表中的主机，如果能找到具有目标IP地址的主机，则将数据包发送给该主机。

3）如果匹配主机失败，则继续搜索路由表，匹配同子网的路由器，如果找到匹配的路由器，则将数据包转发给该路由器。

4）如果匹配同子网的路有器失败，则继续搜索路由表，匹配同网络的路由器，如果找到匹配的路由器，则将数据包转发给该路由器。

5）如果以上匹配操作都失败，就搜索默认路由，如果默认路由存在，则按照默认路由发送数据包，否则丢弃数据包。

从以上路由器的处理步骤可以看出，IP协议并不保证一定把数据包送达目标主机，在发送过程中，会因为数据包结束生命周期，或者找不到路由而丢弃数据包。

（2）域名

尽管IP地址能够惟一标识网络上的主机，但IP地址是数字型的，用户记忆数字型的IP地址很不方便，于是人们又发明了另一种字符型的地址，即所谓的域名（Domain Name）。域名地址具有易于理解的字面含义，便于记忆。IP地址和域名一一对应。例如JavaThinker网站的域名为www.javathinker.org，对应的IP地址为：221.130.187.148。

域名是从右至左来表述其意义的，最右边的部分为顶层域，最左边的则是这台主机的机器名称。域名一般可表示为：主机机器名.单位名.网络名.顶层域名。如：mail.xyz.edu.cn，这里的mail是xyz学校的一个主机的机器名，xyz代表一个学校的名字，edu代表中国教育科研网，cn代表中国，顶层域一般是网络机构或所在国家地区的名称缩写。

DNS（Domain Name System）协议采用DNS服务器来提供把域名转换为IP地址的服务。DNS服务器分布在网络的各个地方，它们存放了域名与IP地址的映射信息。用户需要访问网络上某个主机时，只需提供主机的直观的域名，DNS协议首先请求地理上比较近的DNS服务器进行域名到IP地址的转换，如果在该服务器中不存在此域名信息，DNS协议再让远方的DNS服务器提供服务。

（3）URL（统一资源定位器）

URL是Uniform Resource Locator的缩写，表示统一资源定位器。它是专为标识网络上资源位置而设的一种编址方式，大家熟悉的网页地址就属于URL。URL一般由三部分组成：

应用层协议：//主机IP地址或域名/资源所在路径/文件名

例如JavaThinker网站的BBS的URL为：http://www.javathinker.org/bbs/index.jsp。其中"http"指超文本传输协议，"www.javathinker.org"是Web服务器的域名，"bbs"是网页所在路径，"index.jsp"才是相应的网页文件。

在URL中，常见的应用层协议还包括ftp和file等，比如：

```
ftp://www.javathinker.org/image/
file:///C:\atomcat\webapps\javathinker\admin.jsp
```

以上file协议用于访问本地计算机上的文件，使用这种协议的URL以"file:///"开头。

3. TCP协议以及端口

IP协议在发送数据包时，途中会遇到各种事情，例如可能路由器突然崩溃，使包丢失。再例如一个包可能沿低速链路移动，而另一个包可能沿高速链路移动而超过前面的包，最后使得包的顺序搞乱。

TCP协议使两台主机上的进程顺利通信，不必担心包丢失或包顺序搞乱。TCP跟踪包顺序，并且在包顺序

搞乱时按正确顺序重组包。如果包丢失，则TCP会请求源主机重发包。

如图14所示，两台主机上都会运行许多进程。当主机A上的进程A1向主机B上的进程B1发送数据时，IP协议根据主机B的IP地址，把进程A1发送的数据送达主机B。接下来TCP需要决定把数据发送到主机B中的哪个进程。TCP采用端口来区分进程。端口不是物理设备，而是用于标识进程的逻辑地址，更确切地说，是用于标识TCP连接的端点的逻辑地址。当两个进程进行一次通信，就意味着建立了一个TCP连接，TCP连接的两个端点用端口来标识。在图15中，进程A1与进程B1之间建立了一个TCP连接，进程B1的端口为80，因此进程B1的地址为：主机B:80。进程A1的端口为1000，因此进程A1的地址为：主机A:1000。每个进程有了惟一的地址，TCP就能保证把数据顺利送达到特定的进程。

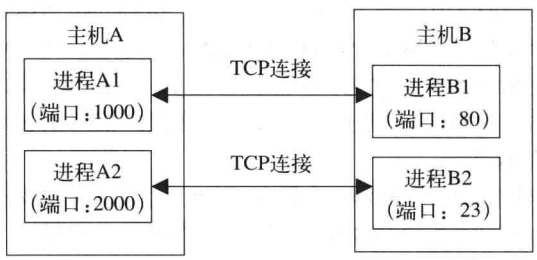


图14　TCP采用端口来区分进程间的通信

端口号的范围为0到65535，其中0到1023的端口号一般固定分配给一些服务。比如21端口分配给FTP服务，25端口分配给SMTP（简单邮件传输）服务，80端口分配给Http（超级文本传输）服务，135端口分配给RPC（远程过程调用）服务等。

从1024到65535的端口号供用户自定义的服务使用。比如假定EchoServer服务使用8000端口。当EchoServer程序运行时，就会占用8000端口，当程序运行结束，就会释放所占用的端口。

客户进程的端口一般由所在主机的操作系统动态分配，当客户进程要求与一个服务器进程进行TCP连接，操作系统为客户进程随机地分配一个还未被占用的端口，当客户进程与服务器进程断开连接，这个端口就被释放。

此外还要指出的是，TCP和UDP都用端口来标识进程。在一个主机中，TCP端口与UDP端口的取值范围是各自独立的，允许存在取值相同的TCP端口与UDP端口。如图15所示，在主机A中，进程A1占用FTP端口1000，进程A2占用UDP端口1000，这是允许的。

4. RFC简介

TCP/IP协议是以RFC（Request For Comment）文档的形式发布的。RFC 是描述互联网相关技术规范的文档。

RFC由个人编写，这些人自愿编写某一新协议或规范的提议草案，并提交给IETF（The Internet Engineering Task Force ，Internet工程任务组织）。IETF负责审阅和发布这些统称为RFC的文档，每个文档都有一个RFC编号，并且处于以下六种类型之一：

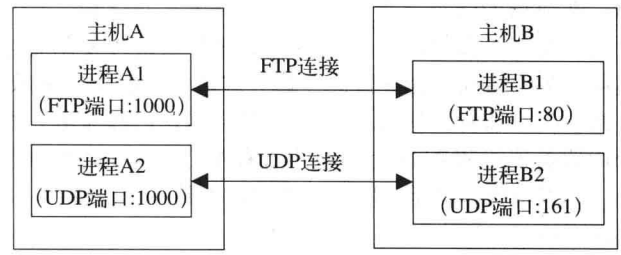


图15　TCP端口与UDP端口的取值范围各自独立

- 标准协议：Internet 的官方标准协议。
- 标准协议草案：正在积极地考虑和审阅以便成为标准协议。
- 标准协议提议：将来可能变成标准协议。
- 实验性协议：为实验目的而设计的协议。实验性协议不投入实际运用。
- 报告性协议：由其他标准组织开发的协议。
- 历史性协议：已经过时的协议，被其他协议代替。

FRC的官方网站为：http://www.ietf.org/rfc.html。在该网站上已经发布了4000多份RFC文档。表2列出了与TCP/IP 协议相关的 RFC文档编号。

表2　与TCP/IP 协议相关的RFC文档编号

RFC 编号	协　　议
768	用户数据报协议 (UDP)
783	日常文件传输协议 (TFTP)
791	Internet协议 (IP)
792	Internet控制消息协议 (ICMP)
793	传输控制协议 (TCP)
821	邮件传输协议（SMTP）
826	地址解析协议 (ARP)
854	Telnet协议 (TELNET)
862	回应协议 (ECHO)
959	文件传输协议 (FTP)
1157	简单网络管理协议 (SNMP)
1939	邮局协议－版本3（POP3）
1945	超级文本传输协议－版本1.0（HTTP1.0）
2060	Internet消息访问协议－版本4（IMAP4）
2068	超级文本传输协议－版本1.1（HTTP1.1）

在FRC的官方网站上输入网址：http://www.ietf.org/rfc/rfcXXXX.txt，就能查看相关的FRC文档，这里的XXXX表示文档编号。例如FTP协议的RFC文档的网址为：http://www.ietf.org/rfc/rfc959.txt。

RFC文档一旦正式发布，其编号和内容就不允许改变。如果需要更新RFC文档，则会对更新后的RFC文档赋予新的编号，再将它发布。例如HTTP1.0协议对应的RFC文档为RFC1945，它的升级版本HTTP1.1协议对应的RFC文档为RFC2068。

5. 客户/服务器通信模式

TCP/UDP协议推动了客户/服务器通信模式的广泛运用。在通信的两个进程中，一个进程为客户进程，另一个进程为服务器进程。客户进程向服务器进程发出要求某种服务的请求，服务器进程响应该请求。如图16所示，通常，一个服务器进程会同时为多个客户进程服务，图16中服务器进程B1同时为客户进程A1、A2和B2提供服务。以下伪代码演示了服务器进程的大致工作流程：

```
while(true){
  监听端口，等待客户请求;
  响应客户请求;
}
```

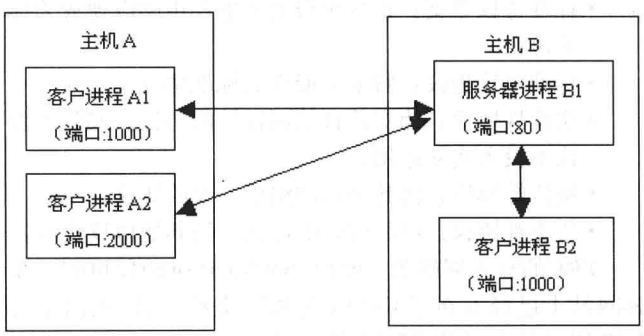


图16 客户进程A1、A2和B2请求服务器进程B1的服务

服务器进程可以提供各种各样的服务，例如本文第一节提到的EchoServer提供的服务为：根据EchoClient发出的字符串XXX，返回字符串“echo: XXX”。除了像EchoServer这样的由用户自定义的服务外，网络上还有许多众所周知的通用服务，最典型的要算Http服务。网络应用层的协议规定了客户程序与这些通用服务器程序的通信细节，例如Http协议规定了Http客户程序发出的请求的格式，还规定了Http服务器程序发回的响应的格式。

在现实生活中，有些重要的服务机构的电话是固定的，这有助于人们方便地记住电话和获得服务，比如众所周知的电话110、120和119分别是报警、急救和火警电话。同样，在网络上有些通用的服务有着固定的端口，表3对常见的服务以及相应的协议和端口做了介绍。

表3 应用层的一些通用服务使用的端口

服　务	端口	协　议
文件传输服务	21	FTP
远程登入服务	23	TELNET
传输邮件服务	25	SMTP
用于万维网（WWW）的超文本传输服务	80	HTTP
访问远程服务器上的邮件服务	110	POP3
互联网消息存取服务	143	IMAP4
安全的超文本传输服务	443	HTTPS
安全的远程登入服务	992	TELNETS
安全的互联网消息存取服务	993	IMAPS

五、用Java编写客户/服务器程序

本文介绍的Java网络程序都建立在TCP/IP协议基础上，致力于实现应用层。传输层向应用层提供了套接字Socket接口，Socket封装了下层的数据传输细节，应用层的程序通过Socket来建立与远程主机的连接以及进行数据传输。

站在应用层的角度，两个进程之间的一次通信过程从建立连接开始，接着交换数据，到断开连接结束。套接字可看做是通信线路两端的收发器，进程通过套接字来收发数据，如图17所示。

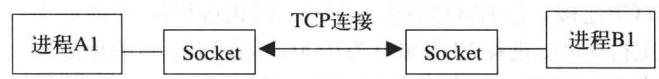


图17 套接字可看过是通信连接两端的收发器

在Java中，有三种套接字类：java.net.Socket、java.net.ServerSocket和DatagramSocket。其中Socket和ServerSocket类建立在TCP协议基础上，DatagramSocket类建立在UDP协议基础上。Java网络程序都采用客户/服务通信模式。

本节以EchoServer和EchoClient为例，介绍如何用ServerSocket和Socket来编写服务器程序和客户程序。

1. 创建EchoServer

服务器程序通过一直监听端口，来接收客户程序的连接请求。在服务器程序中，需要先创建一个ServerSocket对象，在构造方法中指定监听的端口：

```
ServerSocket server=new ServerSocket(8000); //监听8000端口
```

ServerSocket的构造方法负责在操作系统中把当前进程注册为服务器进程。服务器程序接下来调用ServerSocket对象accept()方法，该方法一直监听端口，等待客户的连接请求，如果接收到一个连接请求，accept()方法就会返回一个Socket对象，这个Socket对象与客户端的Socket对象形成了一条通信线路：

```
Socket socket=server.accept(); //等待客户的连接请求
```

Socket类提供了getInputStream()方法和getOutputStream()方法，分别返回输入流InputStream对象和输出流OutputStream对象。程序只需向输出流写数据，就能向对方发送数据；只需从输入流读数据，就能接收来自对方的数据。图18演示了服务器与客户利用ServerSocket和Socket来通信的过程。

与普通I/O流一样，Socket的输入流和输出流也可以用过滤流来装饰。在以下代码中，先获得输出流，然后用PrintWriter装饰它，PrintWriter的println()方法能够写一行数据；以下代码接着获得输入流，然后用BufferedReader装饰它，BufferedReader的readLine()方法能够读入一行数据：

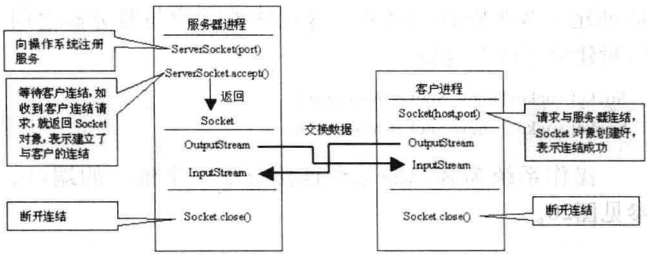


图18　服务器与客户利用ServerSocket和Socket来通信

```
OutputStream socketOut = socket.getOutputStream();
//参数true表示每写一行，PrintWriter缓存就自动溢出，把数据写到目的地
PrintWriter pw=PrintWriter(socketOut,true);
InputStream socketIn = socket.getInputStream();
BufferedReader br=new BufferedReader(new InputStreamReader(socketIn));
```

以下是EchoServer的核心代码。

```
import java.io.*;
import java.net.*;
public class EchoServer {
 private int port=8000;
 private ServerSocket serverSocket;
 public EchoServer() throws IOException {
  serverSocket = new ServerSocket(port);
  System.out.println("服务器启动");
 }
 public String echo(String msg) {
  return "echo:" + msg;
 }
 private PrintWriter getWriter(Socket socket)throws IOException{
  OutputStream socketOut = socket.getOutputStream();
  return new PrintWriter(socketOut,true);
 }
 private BufferedReader getReader(Socket socket)throws IOException{
  InputStream socketIn = socket.getInputStream();
  return new BufferedReader(new InputStreamReader(socketIn));
 }
 public void service() {
  while (true) {
   Socket socket=null;
   try {
    socket = serverSocket.accept(); //等待客户连接
    System.out.println("New connection accepted "
        +socket.getInetAddress() + ":" +socket.getPort());
    BufferedReader br =getReader(socket);
    PrintWriter pw = getWriter(socket);
    String msg = null;
    while ((msg = br.readLine()) != null) {
     System.out.println(msg);
     pw.println(echo(msg));
     if (msg.equals("bye")) //如果客户发送的消息为“bye”，就结束通信
      break;
    }
   }catch (IOException e) {
    e.printStackTrace();
   }finally {
    try{
     if(socket!=null)socket.close(); //断开连接
    }catch (IOException e) {e.printStackTrace();}
   }
  }
 }
 public static void main(String args[])throws IOException {
  new EchoServer().service();
 }
}
```

EchoServer类最主要的方法为service()方法，它不断等待客户的连接请求，当serverSocket.accept()方法返回一个Socket对象，就意味着与一个客户建立了连接。接下来从Socket对象中得到输出流和输入流，并且分别用PrintWriter和BufferedReader来装饰它们。然后不断调用BufferedReader的readLine()方法读取客户发来的字符串XXX，再调用PrintWriter的println()方法向客户返回字符串echo:XXX。当客户发来的字符串为“bye”，就会结束与客户的通信，调用socket.close()方法断开连接。

2. 创建EchoClient

在EchoClient程序中，为了与EchoServer通信，需要先创建一个Socket对象：

```
String host="localhost";
String port=8000;
Socket socket=new Socket(host,port);
```

在以上Socket的构造方法中，参数host表示EchoServer进程所在的主机的名字，参数port表示EchoServer进程监听的端口。当参数host的取值为“localhost”，表示EchoClient与EchoServer进程运行在同一个主机上。如果Socket对象成功创建，就表示建立了EchoClient与EchoServer之间的连接。接下来，EchoClient从Socket对象中得到了输出流和输入流，就能与EchoServer交换数据。

EchoClient的核心代码如下：

```
import java.net.*;
import java.io.*;
import java.util.*;
public class EchoClient {
 private String host="localhost";
 private int port=8000;
 private Socket socket;
 public EchoClient()throws IOException{
  socket=new Socket(host,port);
 }
 public static void main(String args[])throws IOException{
  new EchoClient().talk();
 }
 private PrintWriter getWriter(Socket socket)throws IOException{
  OutputStream socketOut = socket.getOutputStream();
  return new PrintWriter(socketOut,true);
 }
 private BufferedReader getReader(Socket socket)throws IOException{
  InputStream socketIn = socket.getInputStream();
  return new BufferedReader(new InputStreamReader(socketIn));
 }
```

```
public void talk()throws IOException {
 try{
  BufferedReader br=getReader(socket);
  PrintWriter pw=getWriter(socket);
  BufferedReader localReader=new BufferedReader(new InputStream-
    Reader(System.in));
  String msg=null;
  while((msg=localReader.readLine())!=null){
   pw.println(msg);
   System.out.println(br.readLine());
   if(msg.equals("bye"))
    break;
  }
 }catch(IOException e){
  e.printStackTrace();
 }finally{
  try{socket.close();}catch(IOException e){e.printStackTrace();}
 }
 }
}
```

在EchoClient类中，最主要的方法为talk()方法。该方法不断读取用户从控制台输入的字符串，然后把它发送给EchoServer，再把EchoServer返回的字符串打印到控制台。如果用户输入的字符串为“bye”，就会结束与EchoServer的通信，调用socket.close()方法断开连接。

运行范例时，需要打开两个DOS界面，先在一个DOS界面中运行“java EchoServer”命令，再在另一个DOS界面中运行“java EchoClient”命令。图19显示了运行这两个程序的DOS界面。在EchoClient控制台，用户输入字符串“hi”，程序就会输出“echo:hi”。

图19　运行EchoServer和EchoClient程序

在EchoServer程序的service()方法中，每当serverSocket.accept()方法返回一个Socket对象，就表示建立了与一个客户的连接，这个Socket对象中包含了客户的地址和端口信息，只需调用Socket对象的。getInetAddress()和getPort()方法就能分别获得这些信息：

```
socket = serverSocket.accept(); //等待客户连接
System.out.println("New connection accepted "
        +socket.getInetAddress() + ":" +socket.getPort());
```

从图19可以看出，EchoServer的控制台显示EchoClient的IP地址为127.0.0.1，端口为1874。127.0.0.1是本地主机的IP地址，表明EchoClient与EchoServer在同一个主机上。EchoClient作为客户程序，它的端口是由操作系统随机产生的。每当客户程序创建一个Socket对象，操作系统就会为客户分配一个端口。假定在客户程序中先后创建了两个Socket对象，这意味着客户与服务器之间同时建立了两个连接：

```
Socket socket1=new Socket(host,port);
Socket socket2=new Socket(host,port);
```

操作系统为客户的每个连接分配一个惟一的端口，参见图20。

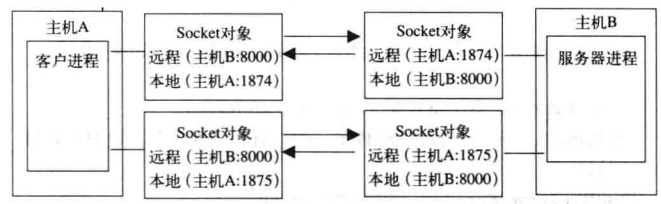


图20　客户与服务器进程之间同时建立了两个连接

在客户进程中，Socket对象包含了本地以及对方服务器进程的地址和端口信息，在服务器进程中，Socket对象也包含了本地以及对方客户进程的地址和端口信息。客户进程允许建立多个连接，每个连接都有惟一的端口。在图20中，客户进程占用两个端口：1874和1875。在编写网络程序时，一般只需要显式地为服务器程序中的ServerSocket设置端口，而不必考虑客户程序所用的端口。

六、结语

计算机网络的任务就是传输数据。为了完成这一复杂的任务，国际标准化组织ISO提供了OSI参考模型，这种模型把互联网络分为7层，分别是物理层、数据链路层、网络层、传输层、会话层、表示层和应用层。每个层有明确的分工，并且在层与层之间，下层为上层提供服务。这种分层的思想简化了网络系统的设计过程。例如在设计应用层时，只需考虑如何创建满足用户实际需求的应用，在设计传输层时，只需考虑如何在两个主机之间传输数据，在设计网络层时，只需考虑如何在网络上找到一条发送数据的路径，即路由。

由于OSI参考模型过于庞大和复杂，使它难以投入到实际运用中。与OSI参考模型相似的TCP/IP参考模型吸取了网络分层的思想，但是对网络的层次做了简化，并且在网络各层（除了主机－网络层外）都提供了完善的协议，这些协议构成了TCP/IP协议集，简称TCP/IP协议。TCP/IP参考模型分为四个层：应用层、传输层、网络互联层和主机－网络层。在每一层都有相应的协议，IP协议和TCP协议是协议集中最核心的两个协议。

IP协议位于网络互联层，用IP地址来标识网络上的各个主机，IP协议把数据分为若干数据包，然后为这些数据包确定合适的路由。路由就是指把数据包从源主机发送到目标主机的路经。

TCP协议位于传输层，保证两个进程之间可靠地传输数据。每当两个进程之间进行通信，就会建立一个TCP连接，TCP协议用端口来标识TCP连接的两个端点。在传输层还有一个UDP协议，它与TCP协议的区别是，