第一天　开放系统、标准及协议

本天介绍一些重要信息，具体包括以下内容

* 什么是开放系统，它是如何处理联网的
* 为什么需要标准，它们是如何开发的
* 协议概述及 OSI 协议细节

1.1　开放系统

这是一本关于 TCP/IP 协议的书。为什么要花时间去考察开放系统和标准呢？这是因为 TCP/IP 协议的产生来自开发标准化通信过程的需要，而这些过程将不可避免地应用到各种平台中。因此，了解一点背景知识有助于更为清楚地认识 TCP/IP 的设计。

更为重要的是，开放系统在当前市场竞争中已成为必不可少的东西。“开放系统”一词作为解决各种问题的方法已盛行于世。理解开放系统的真正含义，有助于更好地了解 TCP/IP 的作用。

同样道理，标准的使用能保证诸如 TCP/IP 之类的协议在各种系统中的一致性。制订标准并非是个简单的过程，它涉及到许多不同协议之间的相互关系。了解 TCP/IP 与通信系统中其他各组成层之间的相互作用，对于正确配置和优化是十分重要的。

如果读者急于开始进入正题，可跳过这些天。但如果这样做，可能会漏过赖以支撑 TCP/IP 的某些基本概念和一些重要专业术语，而这些在后续天中遇到后又必须回头查看，反而麻烦。

1.1.1　什么是开放系统

开放系统有多种定义，但远还没有一个可接受的简明扼要的定义。对于大多数人，开放系统的最佳定义是其体系结构不再成为秘密。关于体系结构的描述可能已公开发行，或对于希望构建平台产品的其他人都很容易得到。这种定义同样适用于硬件和软件。

当多家单一销售商开始生产平台产品时，客户就有了选择余地。用户不是特别喜欢 Nocrash Software 的网络监视软件吗？毫无问题，因为 FaultFree Software 的产品在 Nocrash 硬件上运行，用户一定非常喜欢它的美丽夺目的界面。当然，主要目的是脱离专利平台而走向多家销售商的平台。

十几年前，开放系统实际上还未出现。每个硬件制造商都有自己的产品生产线，用户特别容易限制在所有软件和硬件所需要的制造商上。有些公司利用可捕获的市场，暴利出售，迫使客户接受不希望的配置。反感情绪逐渐膨胀，以致客户开始重视这一问题。

UNIX 是开放软件平台的一个古典实例，它已问世约 30 年。它的源代码不难理解，可用于希望使用它的任何人。UNIX 可以移到许多硬件平台上运行，从而消除了专利的限制。UNIX 的诱人之处并不是操作系统本身特性，而只在于 UNIX 用户可运行其他 UNIX 平台上

的软件，文件是兼容的(除了磁盘格式)，并有各种不同的销售商在出售 UNIX 产品。

UNIX 的发展促使大型硬件制造商转向开放系统原理，致使大多数制造商许可为它们自己的硬件生产 UNIX 系列产品。这一步使客户把不同的硬件系统结合到大型网络上，都运行 UNIX 并一起工作。用户在计算机之间可以几乎透明地移动，而不必考虑他们所在的实际硬件平台。这种开放系统联网最初只是对于最大的公司和政府机构相当重要，现在甚至已成为最小公司的计算机决策的关键考虑因素。

注意：尽管 UNIX 版权现归 X/Open 所有，但已经公布了其操作系统细节。所有想用 UNIX 操作系统来开发应用软件或硬件的开发者都可容易地获得这些细节。UNIX 在这方面是独一无二的。

开放系统联网一词有诸多含义，可谓仁者见仁，智者见智。但很容易看出的一点是，为什么人们需要开放系统联网。有三种广泛使用的服务，它们在网络通信中占有最高的份额：即文件传输、电子邮件及远程登录。

文件传输使用户能快速、有效地共享文件，而不用多余复制或关心传输方法。网络文件传输要比昼夜兼程的通信员要快得多，而且通常要比把文件复制到磁盘上然后把它带出屋子还要快。文件传输还极为方便，这不仅使用户心情愉快，而且也减少了等待材料的时间。

不仅在单个事务领域而且在全球范围内，电子邮件已广泛应用。Internet 承载着政府部门、私人企业、科研院所及私人的成千上万的报文。电子邮件既廉价(不用纸、信封或邮票)，又迅捷(60 秒左右可跨越全球)。它是我们工作所在的基于计算机的世界的明显的扩充。

远程登录能使基于一个系统的用户，通过某个网络与将接受其为用户的任何其他系统建立联系。其他用户可能处于另一个工作组、另一个州或另一个国家。远程登录可使用户充分利用另一地方的特定硬件及软件，并可在另一台计算机上运行应用程序。

1.1.2 体系结构

为了理解联网协议，最好先了解一些有关网络的知识。在阅读本书稍后天中有关路由的内容时，浏览一下此处介绍的最常见的网络体系结构会大有裨益。术语“网络”通常指一组计算机和外围设备(打印机、调制解调器、绘图仪和扫描仪等)，它们通过某种媒介连接在一起。这种连接可以是直接的(通过电缆)，也可以是间接的(通过调制解调器)。网络上的不同外设通过事先定好的一套规则(协议)彼此通信。

所有设备可以放置在同一间房内，也可以分散布置在一栋楼里。通过使用专用电话线路、微波或类似系统，它们甚至可以彼此隔开数里路之远或散布在全球各地，当然，要通过长途通信媒体进行连接。网络的布局(实际设备及它们彼此连接的方式)称为网络拓扑结构。本书着重介绍三种重要的拓扑结构：总线、环及中枢。

总线网络最为简单，它由一条主通信信道及各种设备构成，这些设备在某一处与主通信信道相连。总线网络的一个例子如图 1.1 所示。通路中的每个设备都有特定的编号或地址，其作用是使设备知道进入的信息是针对该设备的。典型的通路是电缆，每台计算机和外设通过连接器连接在电缆上。这样，电缆上的信号就可通过连接器到达设备，而在连接器软件认定报文不是给自己(根据设备的地址)时，信号沿着电缆传输到下一个设备。

总线网络实际上很少是直线电缆，通常要沿着墙壁和建筑物相应地弯曲。从一端到另一端的确只有单一通路，每端都以某种方式结束。图 1.1 说明了网络的逻辑表示，其中把网络

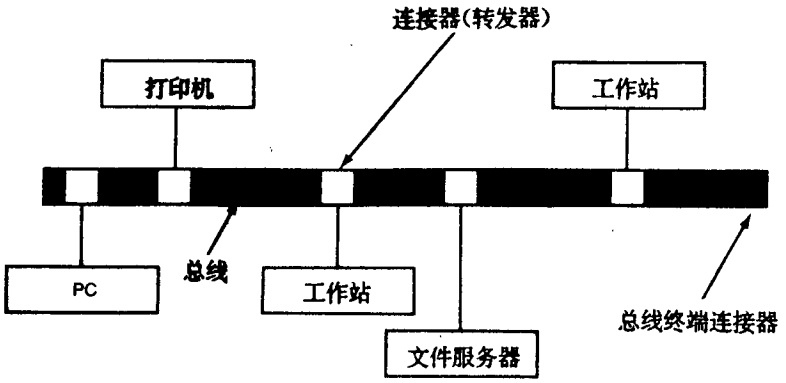

图 1.1　总线网络示意图

的实际物理外形作了简化，用直线连接且未给出连接的实际尺寸，而网络的物理表示将显示出网络如何穿过墙壁，围绕工作台布线等等。总线网络上的大部分设备可以把信息与接收地址结合起来，沿总线发送或接收数据。

环网拓扑顾名思义其形状类似环。典型的环网如图 1.2 所示。术语“环”实际上并不恰当，因为环形网络并不是把总线网络电缆两端连接在一起。相反，环指的是处理网络报文传递的中央单元的设计。

读者以前可能听说过令牌环网，现在发现并没有实际的环形电缆体系结构，可能会有所失望。在令牌环网中，中央控制单元称为媒体访问单元或 MAU。MAU 内部有一环形电路（网络拓扑由此得名）。MAU 内部的环充当设备的总线获取报文。

注意：尽管人们都自然而然地认为环网有一个由两端相连的电缆形成的环形骨架，但实际上电缆环根本就不存在。取名为环源于中央控制单元的结构。

中枢网络使用了一条十分类似于总线网络的主电缆，这条主电缆称为底板。中枢拓扑如图 1.3 所示。从底板引出一套接头至一系列接口，设备藉以安插接头。通至连接点的电缆通常称为“引线”，因为这些电缆从底板引至接口。

中枢网络现在越来越流行，原因之一是它们容易安装和维护。另外的原因是，在许多系统中它是最便宜的。和总线网络一样，底板可以延伸至很远的一段距离，同时，接口或连接点通常组合放在机箱内或面板上。底板上可连接许多面板或连接盒。

1.1.3　层

编写单个软件包去完成不同计算机之间通信所需的每一项任务，是一件骇人听闻的工作。除了不得不处理不同的硬件体系结构外，单为所需要的应用软件编码就会导致程序过长，以致不能执行或维护。

把所有的要求分成组是一个明智的方法，正如程序员把代码分解成逻辑块一样。有了开放系统通信，组是相当明了的。一个组可以处理数据传输，而另一个组可以处理报文的组合，再一个处理终端用户应用程序等等。一组相关的任务就称为“层”。

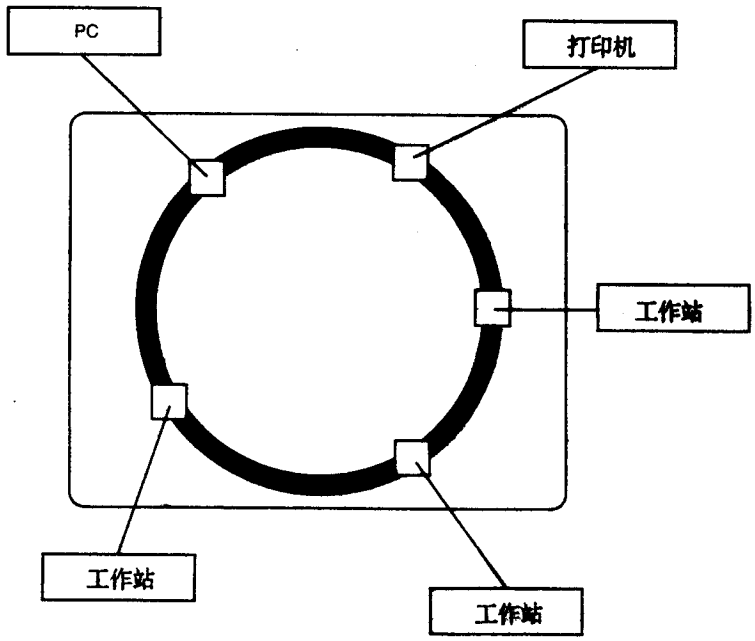

图 1.2 环网示意图

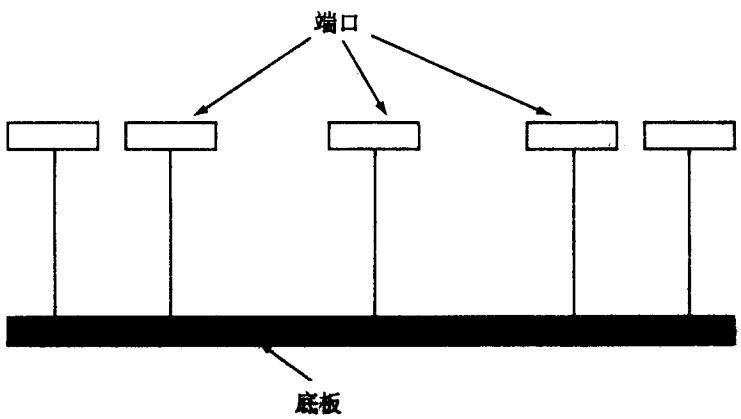

图 1.3 中枢网络示意图

注意：体系结构的各个层都是相对独立的实体，如果不与其他层相互作用，它们通常不能执行任何看得到的任务，但从编程观点看，它们是自包含的。

当然，可以想象到层之间的功能会出现交叉。划分层有几种不同方法，其中可采纳作为标准的是开放系统互联参考模型，在后面的天中将详细讨论。OSI 参考模型(OSI-RM)使用七个层，如图 1.4 所示。TCP/IP 体系结构与此相似，但由于它把某些 OSI 功能进行了压缩，

故仅有五层。现在，让我们来讨论一下这个七层 OSI 模型。

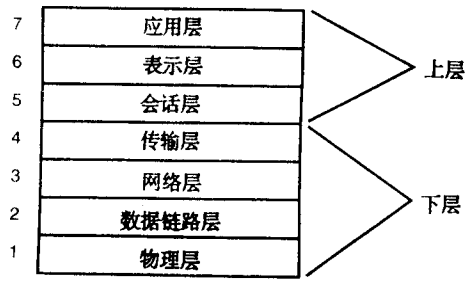

图 1.4　表示所有七层的 OSI Reference Model

应用层、表示层及对话层都是面向应用程序的，它们负责和用户接口，这三层与下面的四层相对独立，而且总体上看，它们和应用程序获取数据的方式无关。

下面四层处理数据的传输，包括每个数据报的组装、路由、验证及传输。这四层并不关心它们接收或发送给应用程序的数据类型，而只负责处理发送数据。它们不以任何方式区分不同的应用程序。

为使读者理解 OSI-RM 的体系结构，下面对每一层进行解释（以后可以对照 TCP/IP 体系结构）。

1.1.3.1　应用层

应用层是 OSI 系统的终端用户界面，是电子邮件、USENET 读者或数据库显示模块等应用程序所在的层。应用层的任务是显示接收到的信息，把用户的新数据发送至较低层。

在如客户—服务器系统之类的分布式应用程序中，应用层是驻留客户应用程序的层，它通过较低层与服务器通信。

1.1.3.2　表示层

表示层的任务是把较低层与应用程序的数据格式隔离开来。它把应用程序中的数据转换成通用格式，这种格式通常称为规范表达式。表示层把来自应用层与计算机有关的数据格式处理成与计算机无关的格式。

文件格式甚至字符格式（例如 ASCII 与 EBCDIC）就是在表示层中去除的。应用程序数据格式的转换是通过“通用网络编程语言”（正如在 OSI 参考模型文档中所称的）而进行的。

表示层对输入数据作相反的处理，它根据应用程序机器指令的类型，把数据从通用格式转换为应用程序专用格式。如果数据进入时没有重新格式化指令，则应用程序就可能以不正确的方式组集输入信息。

1.1.3.3　对话层

对话层组织应用进程之间的数据交换并使之同步。它与应用层协同工作，提供称为“同

步点”的简单数据集，以便让用户知道数据发送与接收的进度。简言之，可把对话层看作是计时与流量控制层。

在协调不同应用程序之间的通信时要涉及对话层，该层使每个应用程序知道其他应用程序的状态。对话层处理应用程序中的某个错误（不论在同一计算机上还是横跨一个国家），以便使接收应用程序知道出现了错误。对话层可使当前正连接的应用程序重新同步化。这在通信暂时中断或发生错误，从而导致数据丢失时是必要的。

1.1.3.4 传输层

根据 OSI 参考模型，顾名思义，传输层就是从源端开放系统到信宿端开放系统的透明数据传输。传输层建立、维护并终止两台计算机之间的通信。

传输层负责确保发送的数据与接收的数据相匹配。传输层检查到错误时要将数据重新发送，因此验证过程对于保证数据正确发送发挥着重要作用。传输层管理数据的发送、确定其顺序及优先权。

1.1.3.5 网络层

网络层提供数据的物理路由，决定计算机之间的路径。网络层处理所有路由问题，以此使较高层不必关心路由问题。

网络层检查网络拓扑，以决定传输报文的最佳路由，同时指出转发系统。只有网络层才真正把报文从信源传输至信宿，并管理通过此系统传输给另外计算机的其他数据块。

1.1.3.6 数据链路层

根据 OSI 参考资料，数据链路层“提供对物理层的控制，检测并纠正可能出现的错误”。实际上，数据链路层的作用是纠正传输过程中出现的传输错误（与应用程序数据自身的错误不同，后者是在传输层中处理的）。

数据链路层通常涉及物理传输媒体上的信号干扰，这些媒体不是铜导线、光纤电缆，就是微波。干扰十分常见，它有许多来源，包括宇宙射线和其他物体的漏磁干扰等。

1.1.3.7 物理层

物理层是 OSI 模型的最低层，根据 OSI 定义，它实现传输数据所要求的机械、电气、功能性及过程手段。这是真正的电线或其他传输形式。

在 OSI 模型形成过程中，焦点集中在下面两层，因为在许多情况下这两层是不可分离的。实际上，人们常把数据链路层与物理层结合在一起，当作一个层，但在正规的 OSI 定义中规定两者有不同的作用（考虑到分层的实用性而非学术性，TCP/IP 把数据链路层与物理层当作一层）。

1.2 术语与符号

对 OSI 与 TCP/IP 都有正规描述，它们是一系列繁琐的文献，定义了协议的各个方面。为了定义 OSI 及 TCP/IP，人们编了一些新词，有些（主要是 OSI 术语）是相当生僻的。有些

OSI 术语指的是这些十分奇异的定义，正如用 legalese 指合法一词一样。

为了更好地理解 TCP/IP 的详细内容，在此有必要介绍一下这些术语。尽管本书中不会出现所有这些术语，但在阅读手册或有关文献时可能会遇到。下面阐述所有主要的术语。

注意：OSI 及 TCP/IP 中的许多术语看上去可能有几种不同的含义，在此试图对每个词给出单一的统一定义。不幸的是，由于用户适应新的术语不是很快，其间可能会引起相当多的混乱。

1.2.1　数据包

实验表明，为了有效传输数据，建立统一的数据块要比发送单个字符或大小变化范围很大的组效果更好。通常在这些数据块前面有一些信息（称为头标），有时在尾部有指示符（称为报尾）。在大部分同步通信系统中把这些数据块称为数据包。

数据包中的数据量和头标的组成随通信协议及某些系统限制而变化，但数据包这一概念总是指的整组数据块（包括头标与报尾）。“数据包”通常在计算机领域中引用，有时也会出现在其他行业中。

警告：“数据包”一般指任何一组包装好的用于传输的数据块。当应用程序数据经过体系结构的各个层时，每一层都在其上增加更多的信息。而“数据包”适用于每一个阶段。通常用这个术语统称任何有附加信息的数据，而不只是某一层附加了头标与报尾以后的特定结果。这与 OSI 与 TCP 体系相悖，但却有利于读者理解。

1.2.2　子系统

子系统是网络中特定层的集合。例如，如果有十台计算机连接在一起，每台计算机运行七层 OSI 模型，所有这十个应用层就是应用子系统。所有十个数据链路层就是数据链路子系统，等等。由此可见，OSI 参考模型共有七个子系统。

子系统中所有单个组成部分在某一时刻为不活动的，是完全可能的。仍以上述十台计算机为例，在任一时刻只有三台计算机的数据链路层实际上在活动，但仍由这十台计算机组成子系统。

1.2.2.1　实体

每一层上可能有多个程序。例如，传输层可能有验证校验和的例程，也可能有处理重发送未正确传输的数据包的例程。由于在任一时刻并不都需要这些例程，所以它们并不总是活动的。这些活动的例程称为“实体”。采用“实体”一词是为了防止与其他计算机术语如模块、进程、任务等混淆。

1.2.2.2　N 表示法

符号(N)、(N+1)、(N+2)等用来标识某一层以及与它有关的层（见图 1.4）。如果传输层是 N 层，则物理层就是 N-3 层，表示层就是 N+2 层。OSI 规定，N 的值总是 1 到 7 之一。采用此表示法的原因之一是为了使作者不必每次都写出其他层的名字，而可用符号取代，同时也使流程图与相互作用图的绘制更为容易。在 OSI 和 TCP 中，N+1 与 N−1 符号都是常见的，分别用于指当前层的上一层与下一层，这一点以后还会看到。

使其更为复杂的是，许多 OSI 标准常用某层名字的第一个字母来表示该层。因为 S-实体，5-实体以及层 5 都指对话层，所以这会给普通读者造成一些混乱。

1.2.2.3 N-函数

每一层都执行 N-函数。函数的作用与层不同。因此，传输层的函数提供的任务是与传输层所提供的不同。本书中函数的含义与实体相同。

1.2.2.4 N-工具

一个层给其较高层提供一套工具，这里用分层结构来表示这个意思。这种表示是明智的，因为应用层希望表示层为它提供一套稳好的、定义良好的工具。用 OSI 语言来讲，(N+1)实体假定使用 N-实体定义了一套 N-工具。

1.2.2.5 服务

提供给(N+1)-实体的一整套 N-工具称为 N-服务。也就是说，服务就是提供给下个较高层的一整套 N-函数。服务看起来类似于函数，但两者形式上不同。OSI 文献通过每层的“服务定义标准”(OSI 这样称呼)用很大篇幅详细描述了服务。在 OSI 标准的形成过程中这种描述是必要的，其作用是把与通信协议有关的不同任务分配给不同的层，使每一层的函数既得到良好的定义又与其他层分开。

服务定义是从底层(物理层)向顶层正式形成的。这种方法的优点是 N+1 层的设计能以 N 层执行的函数为基础，从而避免在相邻的两层出现功能相同的两个函数。

服务名的定义经历了下述变化，有些使用比较规范：

N-服务用户是由 N 层为下一较高层(N+1 层)提供服务的用户。

N-服务提供者是与提供 N 层服务有关的 N-实体的集合。

N-服务访问点(通常简写为 N-SAP)是由 N-服务提供者提供给(N+1)_实体 N-服务的入口。

N-服务数据是在 N-SAP 中交换过的数据包。

N-服务数据单元(N-SDUs)是在 N-SAP 中交换过的单个数据单元(因此 N-服务-数据由 N-SDU 组成)。

这些术语图示于图 1.5。另外一个常用的术语是“封装”，它对数据的数据包附加控制信息。控制数据包含寻址细节、错误检测校验和以及协议控制函数。

1.2.3 术语的理解

在正规叙述中要用到所有这些术语，记住这一点十分重要，因为正规语言通常是正确描述如通信协议这类复杂事情的唯一方法。同时，把这些术语组合起来使其有更多一些意义也是可能的。下面的例子有助于说明这一点。

对话层具有一组对话函数，它给其上层提供了一组对话工具。对话层由对话实体组成，而表示层是由对话层(层 5)提供服务的用户。表示实体是由对话层提供服务的用户，称为表示服务用户。

对话服务提供者是对话实体的集合，这些对话实体实际涉及把对话层的服务提供给表

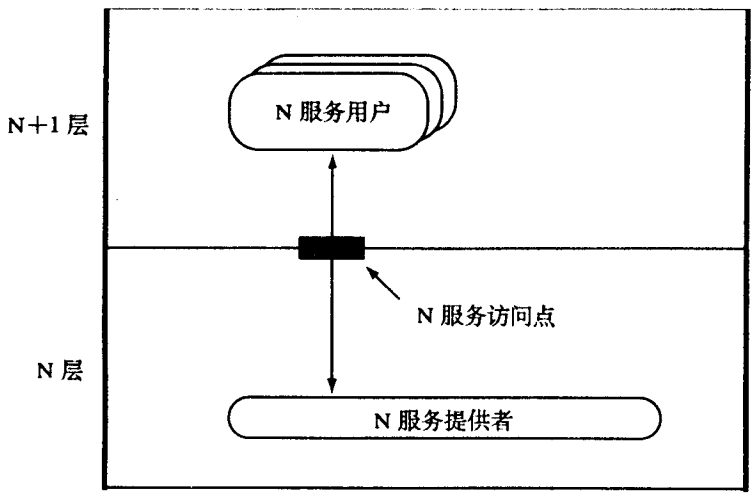

图 1.5　服务提供者通过服务访问点与服务用户通信

示层。把对话服务提供给表示层的入口点是对话服务访问点，在此发送对话服务数据。对话服务数据中的单个位称为对话服务数据单元。

是否比较混乱？可以相信也可不信，读者不久将会感到这些术语是非常合适的。现在需要了解的是，层是通过服务访问点把一组实体提供给下一较高层的，该较高层称为服务用户。数据按服务数据成批发送，服务数据由服务数据单元组成。

1.2.3　队列和连接

双方（无论是通过电话、体系结构中的层与层，还是在若干应用程序自身）之间的通信发生在三个不同阶段：连接建立、数据传输和连接终止。

同一层内两个 OSI 应用程序之间的通信，是通过队列到达它们下面的层的。每个应用程序（确切地应称为服务用户）有两个队列，其中一个队列始终指向下面一层的服务提供者（它控制整个层）。用 OSI 的说法，两个队列在两个 N-服务作用点之间同时（或基本同时）提供相互作用。

数据也称为服务基元，可由应用程序（服务用户）放入队列中，并从队列中检索出。服务基元可以是一个数据块，一表示要求或接收某数据的指示符，或是一个状态指示符。与大多数 OSI 情况一样，可概要描述下列这些队列的作用：

请求基元是当一个服务可以为队列提供一个基本服务要素（通过 N-SAP）时，该要素请求允许与同一层的另一个服务进行通信。

指示基元是位于发送应用程序的下面一层的服务提供者，是向预定的接受应用程序发送的东西，以使该程序知道所期望的通信。

响应基元由接受应用程序发送到该层下面的服务提供者，以确认许可两个服务用户之间的通信。

证实基元从服务提供者处发送最终应用程序，以指示该层上可进行以上通信的两个应

用程序。

举个例子可能有助于澄清这一过程。假设表示层中两个应用程序要相互通信，但它们不能直接(根据 OSI 模式)通信，而必须通过它们下面的层，这些步骤如图 1.6 所示。

第一个应用程序将请求基元发送到对话层的服务提供者然后等待。该对话层服务提供者将从第一个应用程序的入站队列中删除请求基元，并向第二个应用程序的入站队列发送指示基元。

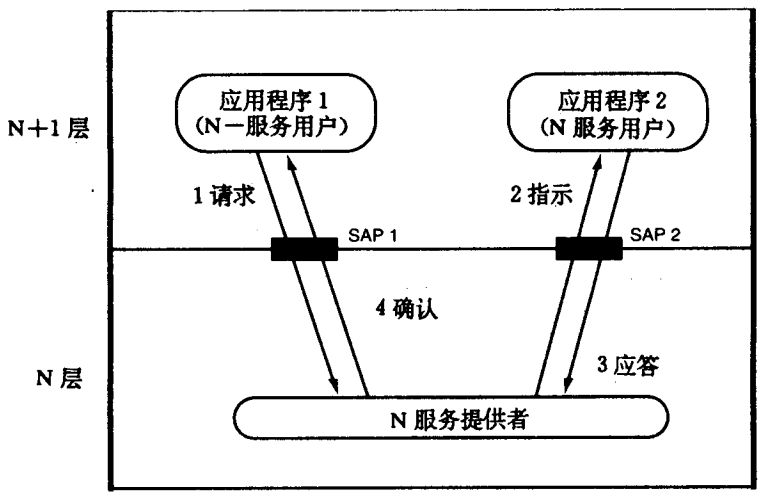

图 1.6 两个应用程序用原语通过 SAP 通信

第二个应用程序从其队列取出指示基元，发向对话服务提供者，并通过其队列将肯定响应基元发送回对话层，以决定是否接受连接请求。对话层服务提供者接收该请求，并将证实基元发送到表示层中的第一个应用程序。此过程称为证实服务，因为应用程序等待证实建立并准备通信。

OSI 还提供未证实服务，这种服务有一个发送到服务提供者的请求基元，OSI 将指示基元发送到第二个应用程序，但不发送响应基元和证实基元。这是一种“即时可用，与用户无关”的通信，通常称为发送请求。

当两个服务用户同时使用证实服务通信时，即认为已经连接成功。两个应用程序可相互对话，并可了解对方正在用服务数据所进行的内容。OSI 指示是在两个应用程序之间建立并维护状态信息，或各自都知道对方是在发送还是在接收，OSI 将这种功能称为面向连接的或连接模式的通信。

无连接通信是指服务数据可独立发送，如未证实服务。服务数据是自包含的，并处理接收服务用户需要知道的任何事情，这些服务数据包常常称为数据报。发送数据报的应用程序不知道是谁接收数据报以及数据报如何处理，而接收服务用户不知道是谁发送(除了可在数据报自身内部包含的信息)。OSI 称之为无连接模式。

OSI(以及 TCP/IP)在其体系结构层之间既使用连接的又使用无连接的系统。每个系统都有其自身的优点和理想的产品，所有这些通信都是在每个层的应用程序(服务用户)之间

进行的，用其下一层进行通信。一般情况下有很多服务用户，该过程始终在进行。当用户仔细思考这一点时，会发现这是相当惊人的。

1.3 标　　准

毫无疑问，玩纸牌游戏需要有一套规则。假如没有规则，每个人可能会采用自己喜欢的玩法，而不管是否与其他人协调一致。规则的存在可确保每个人按相同的方法参与游戏，这可能不如我行我素要有意思，然而在发生争吵时，已写明的规则将表明谁是对的。规则就是游戏所遵循的一组标准。

标准用来防止产生这种情况：两个看起来似乎兼容的系统其实不兼容。如十年前CP/M是占有主导地位的操作系统，大多数系统都采用软件但都没有5.25英寸软盘。但是Kaypro I的软盘无法用Osbourne I读出，因为两种系统对软盘磁道的划分格式不同。尽管利用实用程序可以将两种格式进行相互转换，但这额外的操作（步骤）会使用户感到烦恼。

在IBM PC成为主导工作平台后，其他公司也采用IBM PC使用的5.25英寸软盘格式，从而确保软盘的兼容性。由于市场压力和用户需求，IBM格式已成为事实上的标准。

1.3.1 制定标准

在当今世界，创建一套标准并非易事。一些组织致力于开发完整的无歧义性的标准。这些组织中最重要的要属国际标准化组织或ISO（常缩称为International Standardization Organization，尽管这并不正确）。ISO由许多国家的标准化组织构成，其成员包括美国国家标准协会（ANSI），英国标准协会（BSI），德国工业标准协会（DIN）和法国标准协会。ISO制定的开放系统互联（OSI）标准将贯穿全书。

当然，每个国家的标准化组织都可制定适合本国国情的标准。而ISO的宗旨是制定符合全球范围的标准。换句话说，不一致性的存在将会导致一个国家的系统不能在其他国家使用（如电视信号，美国使用NTSC制式，而欧洲使用PAL制式，这些制式互不兼容）。

很巧，大多数国际标准所使用的语言是英语，但标准委员会中的许多成员都是来自非英语国家，这将会产生许多混淆，特别是当许多标准是用难以阅读的方式写成的情况下。

大部分标准使用难以理解的术语写成，原因是因为无歧义的描述有时可能非常困难，有时必须创建标准定义新条目。不仅必须清楚定义概念，而且还必须绝对有效。对于标准适用的大多数问题，这意味着使用数字和物理术语来提供具体定义。定义一个2×4的小块只需要使用某些度量，而类似地定义计算机术语必须使用数学知识。

简略地定义通信方法，像TCP/IP，而不用对计算系统进行复杂定义，将会变得十分简单明了。开放系统的使用将增加另一个困难，因为所有标准内容必须与设备无关。设想一下不用我们所熟悉的测量单位（如英寸）来定义2×4小块将会是什么样子；而即使用英寸来测量，无歧义地定义英寸也仍将是比较困难的事情。

计算机通信是通过数据位实现的，这些数据位可以表示字符、数字或其他任何符号。数字可以是整数、分数或八进制数。同理，我们必须定义最基本单位，读者可以看到，复杂性升级了，一个比一个复杂。

使用抽象方法有助于定义标准。如OSI要首先处理传输数据的含义（称为语义），但另

一方面，计算机中数据的确切表示（实际语法）和传输方法（传输语法）是单独处理的。这种概念上的区分使数据可按整体表示，而与它实际的含义无关。这有点像把汽车作为单位而不是发动机、变速器和方向盘等。把具体东西抽象为单一的整体，可使信息传输变得十分容易（"汽车坏了"是抽象的，而"油箱泄漏"是具体的）。

按抽象方式描述系统需要有某种符合语言。大部分标准群体已开发了这样一个系统，最常用的是 ISO 的抽象语法表示 1，通常缩写为 ASN.1。它尤其适合于描述开放系统联网。这样，在 OSI 和 TCP 的描述中广泛使用 ASN.1 并不令人惊奇。实际上，在需要描述高层功能时，ASN.1 与 OSI 标准是同时开发的。

ASN.1 的主要概念是所有数据类型，不论其类型、大小、来源和用途，它们都可以用硬件、操作系统和应用程序的对象来表示。ASN.1 系统定义数据报协议头标的内容，头标是描述系统内容的对象开头的信息块（在本天"协议头标"一节中将对头标作具体讨论）。

ASN.1 一部分描述用于描述对象和数据类型（就像用数据库术语中的数据描述语言）的语言。另一部分定义基本的编码规则，该规则用来处理在两个系统之间移动数据对象。ASN.1 定义数据包（数据报）结构中所使用的数据类型，它提供了二十八种支持结构化和非结构化数据类型。

注意：本书没有过多考虑介绍 ASN.1，只是两个地方顺便提及它。然而了解 TCP/IP 所有方面的一般定义语言是很有用的。

1.3.2 Internet 标准

在 1980 年国防部高级研究计划署成立时，组建了 Internet 标准开发小组。该小组（称为 Internet 配置控制委员会，ICCB）在 1983 年并入了 Internet 指导委员会（IAB），其任务是设计、规划和管理 Internet。

1986 年，IAB 把开发网际标准的任务交给了 Internet 工程特殊任务组（IETF），把长期研究任务交给了 Internet 研究特殊任务组（IRTF）。IAB 保留对两个机构建议的所有事务的最终裁决。

该阶段最后一步是 1992 年 Internet 学会的成立，其中 IAB 改名为 Internet 结构委员会。该委员会仍然负责已有的和未来的标准，并负责向 Internet 学会委员会汇报工作。

通过改组，最后发生了什么事情呢？几乎开始就把 Internet 定义为"组织松散的独立国际合作的互联网络"，"通过自主遵守计算的协议和过程"支持主机对主机的通信，是在 Internet 标准技术文件 RFC 1310.2 中定义的。该定义至今仍在使用。

IETF 继续从事提炼标准的工作，该标准用来在 Internet 间通过许多工作组进行通信，每个工作组用于完成整个 Internet 协议程序集的特定部分。目前已有致力于网络管理、安全性、用户服务、寻径和其他许多内容的工作组。有意思的是 IETF 委员会比 ISO 更加灵活和有效，因为 ISO 工作组需花费时间来对标准征求意见。在许多情况下，IETF 委员会可形成、产生一个建议，并在年内解散，这有助于不断地完善 Internet 标准，从而反映硬件和软件的性能变化。

建立新的 Internet 标准（TCP/IP 所发生的）要遵循一系列过程，如图 1.7 所示。从评议征求（RFC）开始，它通常是个包含具体建议，有时是新的有时是对已有标准的修改意见。RFC 分布广泛，不但网络上有，而且还向感兴趣的部门发行印刷品。

RFC 通常讨论网络自身的一些问题，在此可各抒己见，这在正式的 IETF 工作委员会会议上也是如此。在适当数量的修正及继续讨论后，制定 Internet 草案，然后到处发布。这个草案接近于最终形式，它总结了 RFC 中的所有建议。

下一步是提出标准，这一阶段至少需要六个月。在这期间 Internet 学会要求至少开发和测试两套独立的产品，在实际测试中发生的任何问题随后便可以解决（实际上，通常要写许多软件并进行详细测试。）

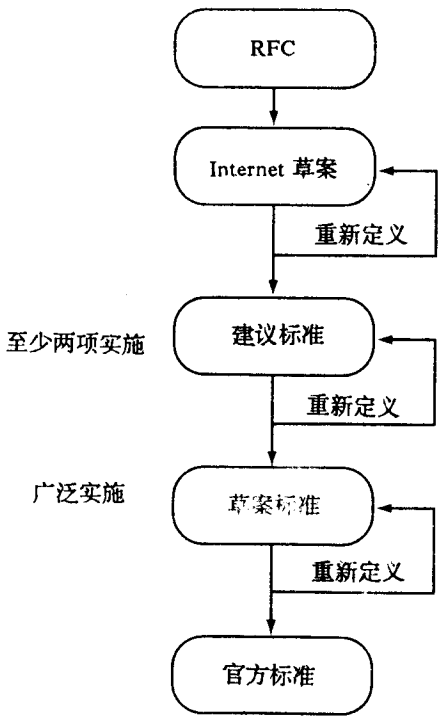

图 1.7　采用一个新的网际标准的过程

在测试和提炼过程完成后，写成标准草案，该标准至少保留四个月。在此期间开发和测试更多的产品。最后一步就是实施在许多月以后的标准，至此就可将它提供给所有需要它的站点去执行。

1.4　协　　议

外交官在磋商国家事务时，要遵循一套规则，此处我们可以把它们看作为一种协议。外交礼仪要求不能侮辱主人并且入乡随俗，多数大使馆有专门的礼仪人员，他们的职责是当双方交往发生磨擦时确保任何事情都能得到圆满解决。礼仪是一套规则，就像职业外交官常说的，这套规则是“玩游戏”必须遵守的。如果没有协议，对话双方就可能不明白对方所说的话。

同样，计算机协议定义实现通信的方式，如果一台计算机正在向另一台计算机发送信息，并都严格遵守协议，信息会顺利传输，而不管计算机是什么类型，也不管它们运行什么操作系统（开放系统的基础）。只要计算机有能够管理协议的软件，通信就可以实现。实际上，

计算机协议只不过是协议信息交换的一组规则。

协议是从非常简单的过程(一方发送一字符,对方把该字符送回来,确保二者匹配)到能解决所有可能问题和传输条件的精确、复杂的机理而开发出来的。像从一个海岸向另一个海岸发送报文这样的任务,当考虑信息传输的方式时,通信将会变得非常复杂。包括所有传输定义的协议将会变得过于庞大,过分专业化。因此,人们开发了许多协议,而每一个协议只能解决一个特定问题。

若把一系列具有既定用途的协议综合起来,假如没有明确地定义这些协议之间的相互作用,那将是非常盲目的。分层结构概念的提出有助于定义每个协议的功能,并且可定义协议之间相互作用的方式(实际上,通信协议就在协议之间)。

如前所述,国际标准化组织(ISO)开发了分层协议系统,称为开放系统互联(OSI)。OSI 把协议定义成一组规则和格式(语义和语法),它确定(N)实体使用(N)函数所体现的通信方法。N 代表层,而实体是层的服务基元。

在设备通信时,可形式地定义规则,并解决信息流中可能出现的中断或错误,特别是当信息流是无连接时(两台设备间不存在形式上的连接)。在这样的系统中,正确寻径和验证每个数据分组(数据报)是非常重要的。就像前面所讨论的,两层之间的数据传输称为服务数据单元(SDU),因此,OSI 把两台设备之间类似的数据定义为协议数据单元(PDU)。

信息流由一组定义协议状态设备的操作来控制,OSI 把这些操作定义为协议控制信息(或 PCI)。

1.4.1 数据分割

介绍更多一些在 OSI 和 TCP 中经常使用的术语是必要的。然而幸运的是,由于这些术语的客观含义,所以很易于理解。因为数据常常不是以可管理的块存在的,所以这些术语是必要的。数据可能不得不分成若干比较小的部分,或把若干小的部分再组合在一起构成大的数据段,以便于更有效地传输。基本术语包括:

分段:是把 N-服务数据单元(N-SDU)分成若干 N-协议数据单元(N-PDU)的过程。

重组把若干 N-PDU 组合成 N-SDU 的过程(是分段的逆向操作)。

组块:在产生 SDU 的层内把若干 SDU(可能来自不同的服务)组合成较大的 PDU。

分块:在同一层内把 PDU 分解成若干 SDU。

联结:是一种把下一个较高层的若干 N-PDU 组合成一个 SDU 的过程(除跨越层边界外,与组块相似)。

分离:是联结的逆过程,为其下一较高层把单个 SDU 分解成若干 PDU(除跨越层边界外,与分解块相似)。

上述六个过程如图 1.8 所示。

最后,介绍用于处理连接的一组定义:

多路复用:是指下一较低层的单一连接支持当前层的若干连接(如三个图像服务连接可复用单一对话连接)

多路分离:是多路复用的逆过程,它为其上层把一个连接分成若干连接。

分解:是单个连接由较低层的若干连接支持(因此数据链路层可用三个连接支持一个网络层连接)。

重组：是分解的逆过程，为较上层把若干连接组合成单个连接。

多路复用和分解（及它们的逆过程：多路分离和重组）在线路分离形式上是不同的。对多路复用而言，是把若干个连接组合成低层的一个连接；而对于分解，是把一个连接分解成低层的若干连接。毋庸置疑，在 TCP 和 OSI 中，它们中的每一个都很重要。

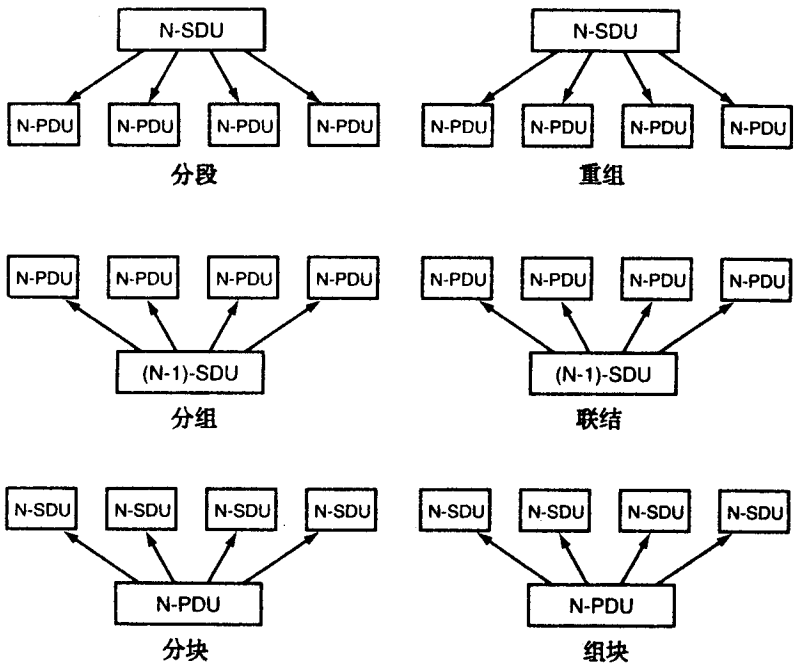

图 1.8　分段、重组、组块、分块、联结、分离

1.4.2　协议头标

协议控制信息是关于所属数据报的信息，这个信息通常组成一个块，附加在数据的前面，称为头标或协议头标。头标用于在不同层之间传输信息，也用于设备之间传输信息。如前所述，协议头标是根据 ISO 的抽象语法表示文档集中设计的规则而开发的。

当协议头标传输到下一层时，对接收层而言，含有层头标的数据报是作为完整数据报来处理的，接收层把自己的协议头标附加在数据报的前面。这样的话，如果一个数据报从应用层出发，在到达物理层时，在其上面将附加有七个协议头标。这些层的协议头标在返回上一层时使用，逐层把头标剥去，然后将数据报送给上一层，如图 1.9 所示。

不难想象，这个过程就像剥洋葱皮，内部是要传输的数据。当数据报经过 OSI 模型的每一层时，附加上一层洋葱皮。完成传输时，有若干个协议头标把数据包围起来。数据回送时（通常由另一台设备完成），每一层剥去它自己的协议头标，当到达目标层时，只剩下数据部分。

设计这样一个过程的是很明智的，因为 OSI 模型的每一层都要求从数据报获取不同信息。对数据报各层，使用特定的协议头标，这样去除协议头标、对其指令进行解码，然后把的

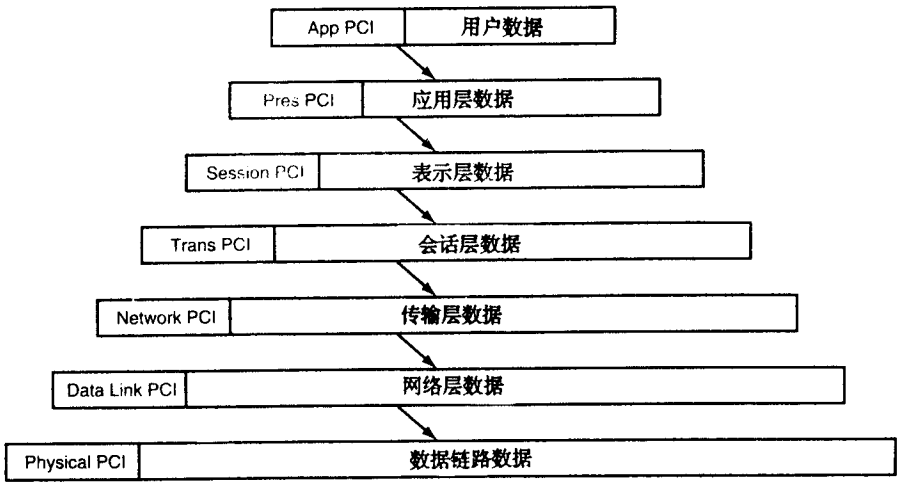

图 1.9 增加每一层的 PCI 到用户数据

其余部分往上传，就成了一件相对较容易的事情。反之，假如只有一个大型头标包含所有信息，将会花费较长时间来处理这些事情。协议头标的确切内容现在还并不重要，本书将在稍后讲述 TCP 协议时再作讨论。

通常，OSI 对所有这些都有正规描述，其中规定了用 N-协议控制信息(N-PCI)来修饰要传输的 N-用户数据，从而形成 N-协议数据单元(N-PDU)。N-PDU 作为组成 N-服务数据单元的一组服务参数越过 N-服务访问点(N-SAP)。组成 N-SDU 的服务参数称作 N-服务用户数据(N-SUD)，它可追加到(N-1) PCI 以形成另一个(N-1)PDU。

对层内的每一个服务，都有与其下层通信的协议(注意：应用程序—服务—通过低层进行通信，但不是直接的)。每项服务的协议转换是由系统定义的，应用程序开发者通常只遵守系统的这些规则，而极少处理协议转换。

协议和协议头标的概念听起来已比较复杂，它们所要完成的工作就更为复杂了。但是，就 OSI 模型的初衷而言，这已是最佳方案了(曾有很多人指责 OSI 和 TCP，说协议头标信息比要传输的数据内容还要重要)。这在某种意义上的确如此，因为没有协议头标，数据就无法传输到信宿。

1.5 小 结

本天列举了许多术语，它们在以后的天中会经常遇到，读者了解这些术语会有很多好处。然而，更深一步了解它们之间的关系，还需时常返回到本天，再回顾一下各种术语的意义。

至此，读者已具备了把 TCP/IP 和 OSI 分层模型联系起来的基本知识，这有助于理解 TCP/IP 是做什么的(以及它是如何完成的)。在下一天，我们将回顾一下 TCP/IP 的发展史和 Internet 的发展。

1.6　问题(Q)与答案(A)

Q　网络体系结构有哪三种主要形式？它们的主要特点是什么？

A　三种体系结构为：总线结构、环形结构和中枢结构，还有其他类型的网络体系结构，但大多数局域网都采取这三种形式之一。

总线网络中所有微机及外围设备通过连接器接到一根电缆上，电缆有两端点。环形网络中所有设备通过电缆连接到一个中央控制单元上，中央控制单元又称为媒体访问单元。中枢网络中每一个设备都通过电缆连接到一个底板的各个连接器上。

Q　OSI 分层模型有哪七层？并用一个短语概括说明各层作用。

A　OSI 各层(自下向上)依次为：

物理层：传输数据

数据链路层：纠正传输错误

网络层：提供物理路由信息

传输层：验证数据已正确传输

对话层：使上下层之间的数据交换同步化

表示层：将网络数据转换为应用软件专用格式

应用层：终端用户界面

Q　分段和重组有什么区别？联结和分离又有什么区别？

A　分段是指把一个大型 N-服务数据单元(N-SDU)分解成几个较小的 N-协议数据单元(N-PDU)，而重组是分段的逆过程。

联结是指将来自上一层的几个 N-PDU 组合成当前层的一个 SDU，分离是联结的逆过程。

Q　什么是多路复用和多路分离？用途是什么？

A　多路复用是指单一连接支持若干连接。根据正规定义，多路复用适用于各层(因此三个表示服务连接可多路复用到单一对话连接上)。但这个词适用于所有类型的连接，如将四个调制解调器连接到一条调制解调器线上。多路分离是多路复用的逆过程，即将一个连接分成数个连接。

多路复用的意义在于通过有限的信源支持多个连接，典型的例子如：带有二十个终端的远程办公室每个终端通过电话线与主办公室连接。但实际上，不需要二十条电话线，而只要三或四条就可以了，这就需通过多路复用方式来连接。线的数量是根据每一条物理线的最大容量来决定的。

Q　在基于 OSI 的电子邮件应用程序(在应用层)已把报文传输到物理层时，其上附加了多少个协议头标？

A　七个，每一层都附加一个协议头标，在实际的物理网络系统中，附加的协议头标更多。一般来讲，每一层都加上它自己的协议信息。