计算机网络（第 8 版）

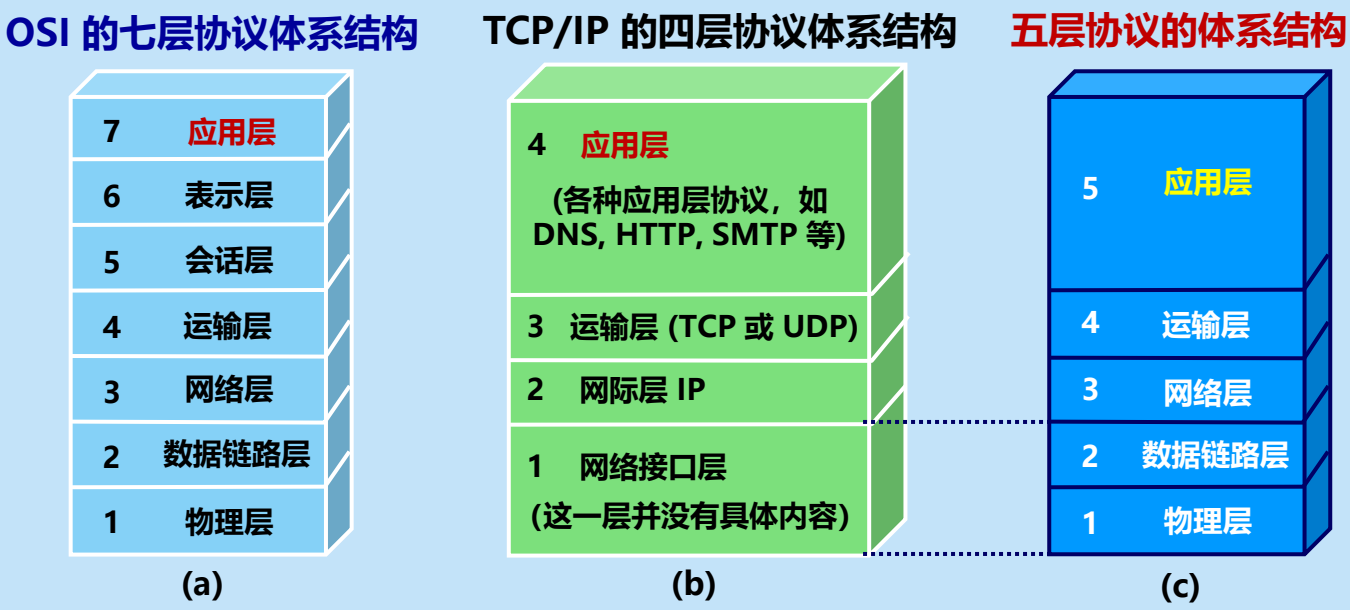
谢希仁 编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

第 6 章

应用层

计算机网络体系结构


OSI 的七层协议体系结构
7 应用层
6 表示层
5 会话层
4 运输层
3 网络层
2 数据链路层
1 物理层
(a)
TCP/IP 的四层协议体系结构
4 应用层
(各种应用层协议，如 DNS, HTTP, SMTP 等)
3 运输层 (TCP 或 UDP)
2 网际层 IP
1 网络接口层
(这一层并没有具体内容)
(b)
五层协议的体系结构
5 应用层
4 运输层
3 网络层
2 数据链路层
1 物理层
(c)

6.1 域名系统 DNS
6.2 文件传送协议
6.3 远程终端协议 TELNET
6.4 万维网 WWW
6.5 电子邮件
6.6 动态主机配置协议 DHCP
6.7 简单网络管理协议 SNMP
6.8 应用进程跨越网络的通信
6.9 P2P 应用

应用层协议

- 精确定义不同主机中的多个应用进程之间的通信规则。
- 包括：
  - 应用进程交换的报文类型，如请求报文和响应报文。
  - 各种报文类型的语法，如报文中的各个字段及其详细描述。
  - 字段的语义，即包含在字段中的信息的含义。
  - 进程何时、如何发送报文，以及对报文进行响应的规则。

应用层协议

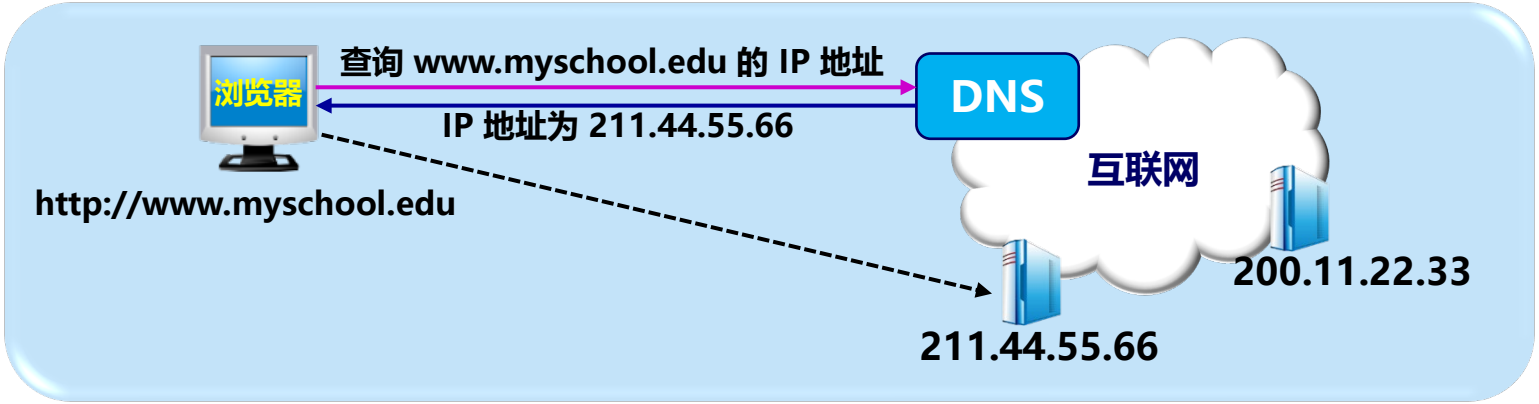
不同于网络应用		许多都基于客户服务器方式	
万维网	HTTP	客户 (client)	服务器 (server)
电子邮件	SMTP，POP3	服务请求方	服务提供方
文件传输	FTP	应用进程	应用进程

6.1 域名系统 DNS

6.1.1 域名系统概述
6.1.2 互联网的域名结构
6.1.3 域名服务器

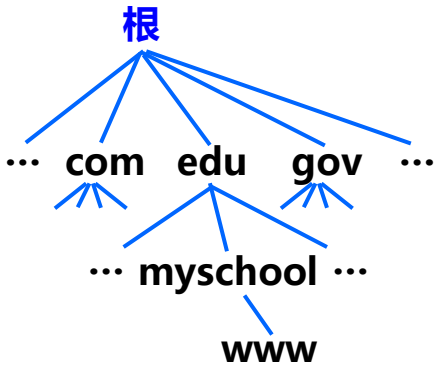
6.1.1 域名系统概述

- **域名系统 DNS (Domain Name System)：**
  - 互联网使用的命名系统。
  - 用来把人们使用的机器名字（域名）转换为 IP 地址。
  - 为互联网的各种网络应用提供了核心服务。

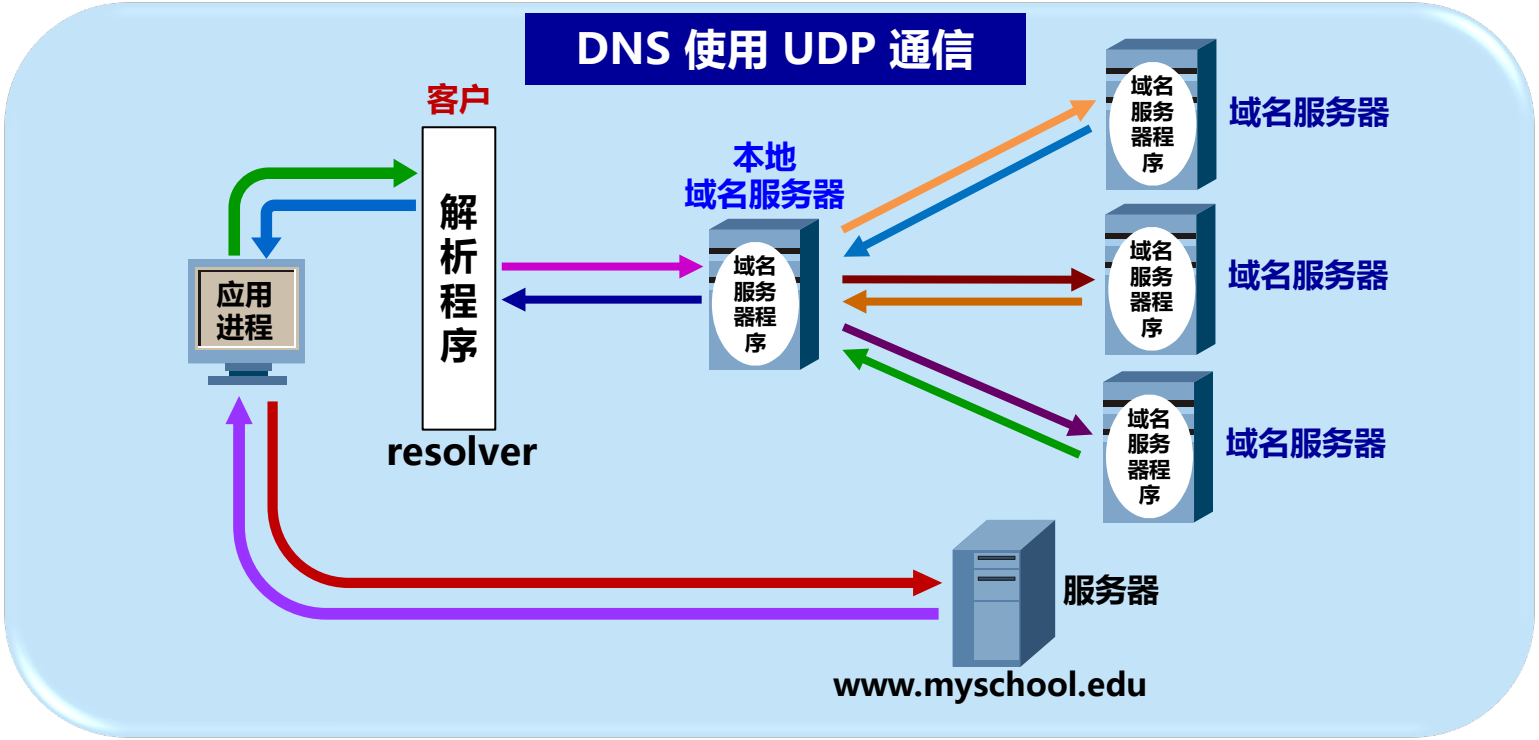


6.1.1 域名系统概述

- 域名采用层次树状结构的命名方法：www.myschool.edu。
- DNS 是一个联机分布式数据库系统，采用客户服务器方式。
- 域名到 IP 地址的解析是由若干个域名服务器程序共同完成。
- 域名服务器程序在专设的结点上运行，运行该程序的机器称为域名服务器。



域名解析过程要点


DNS 使用 UDP 通信
客户
解析程序
resolver
应用进程
本地域名服务器
域名服务器程序
域名服务器
域名服务器程序
域名服务器
域名服务器程序
域名服务器
服务器
www.myschool.edu

6.1.2 互联网的域名结构

- **命名方法：**层次**树状**结构方法。
- 任何一个连接在互联网上的主机或路由器，都有一个**唯一**的层次结构的名字，即**域名** (domain name)。
- **域 (domain)：**
  - 名字空间中一个**可被管理**的划分。
  - 可以划分为**子域**，而子域还可继续划分为子域的子域，这样就形成了顶级域、二级域、三级域，等等。

6.1.2 互联网的域名结构

- **域名结构：层次结构。**由标号 (label) 序列组成，各标号之间用点（.）隔开，各标号分别代表不同级别的域名。

注意：域名只是个逻辑概念，并不代表计算机所在的物理地点。

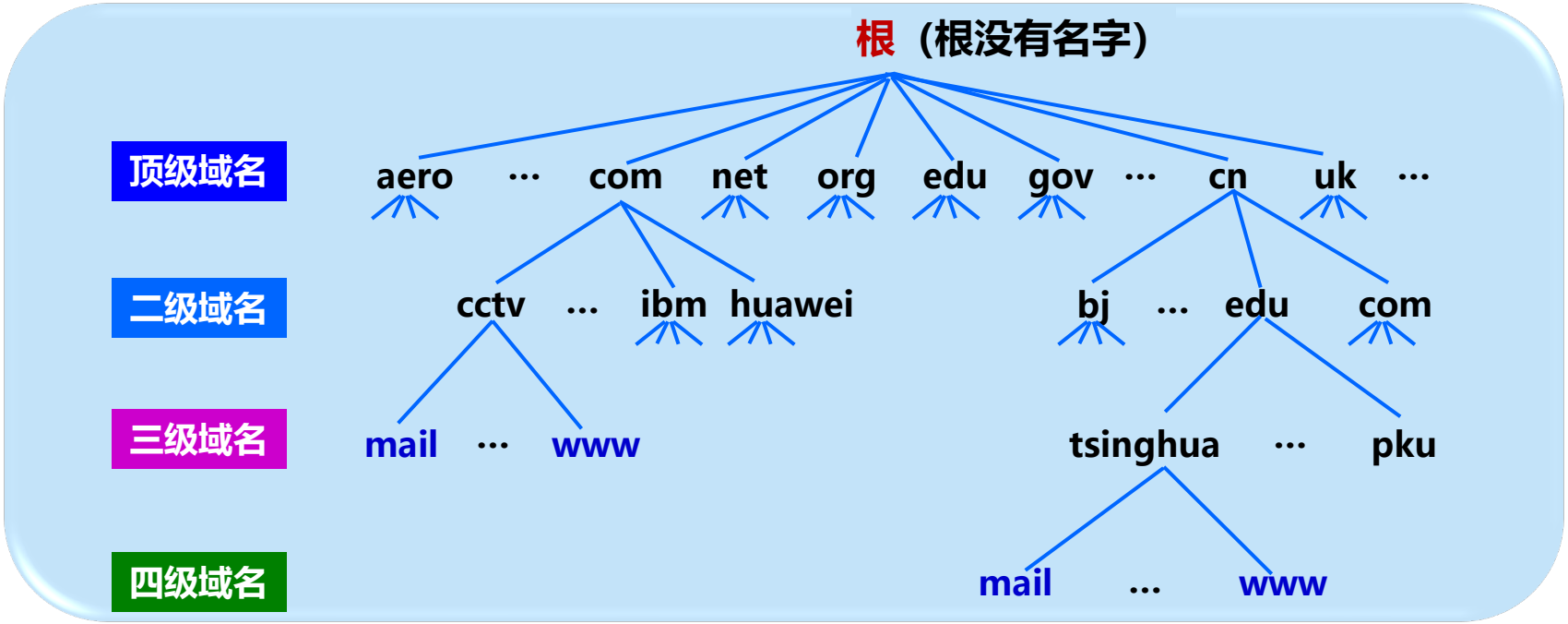
全球顶级域名 TLD (Top Level Domain)

国家顶级域名 nTLD	通用顶级域名 gTLD	基础结构域名 (infrastructure domain)	新顶级域名 (New gTLD)
采用 ISO 3166 的规定	com，net，org，edu，gov 等	只有一个：arpa	任何公司、机构都有权向 ICANN 申请新的顶级域
又常记为 ccTLD	总数已达 20 个	用于反向域名解析，又称为反向域名	真正的企业网络商标
总数已达 316 个			

在国家顶级域名下注册的二级域名均由该国家自行确定。

我国把二级域名划分为“类别域名”和“行政区域名”两大类。

互联网的域名空间结构



域名树的**树叶**就是计算机的名字，它不能再继续往下划分子域了。

互联网的域名空间结构

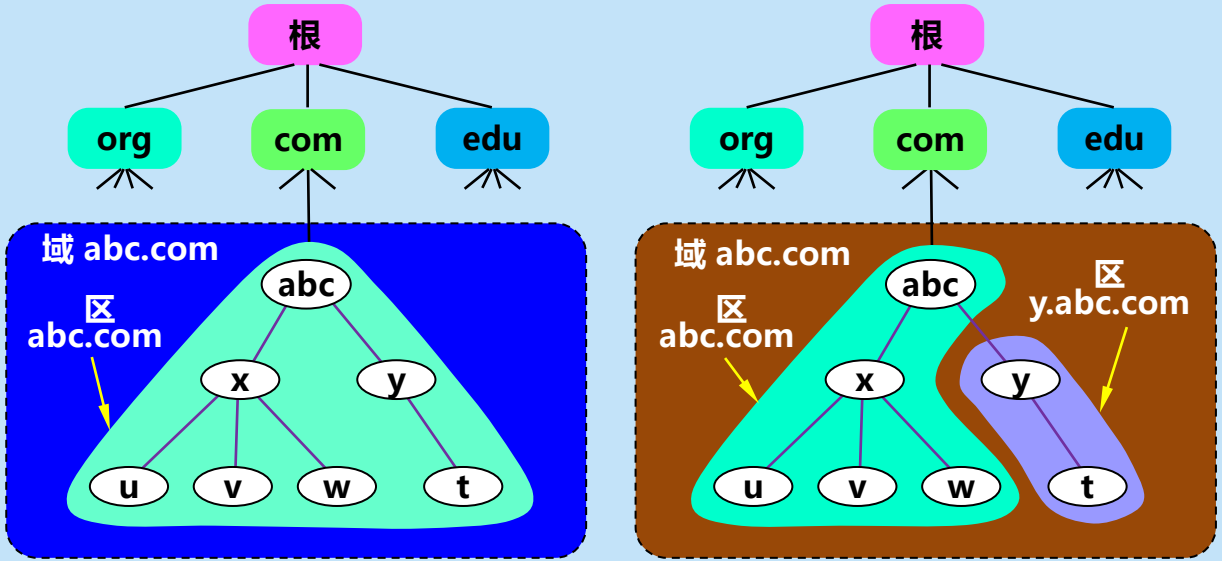
注意：互联网的名字空间是按照机构的组织来划分的，与物理的网络无关，与 IP 地址中的“子网”也没有关系。

6.1.3 域名服务器

- 实现域名系统使用分布在各地的**域名服务器**（DNS 服务器）。
- 一个服务器所负责管辖的（或有权限的）范围叫做**区** (zone)。
- 各单位根据具体情况来划分自己管辖范围的区。但在一个区中的所有节点必须是能够**连通**的。
- 每一个区设置相应的**权限域名服务器**，用来**保存**该区中的所有主机的域名到 IP 地址的映射。

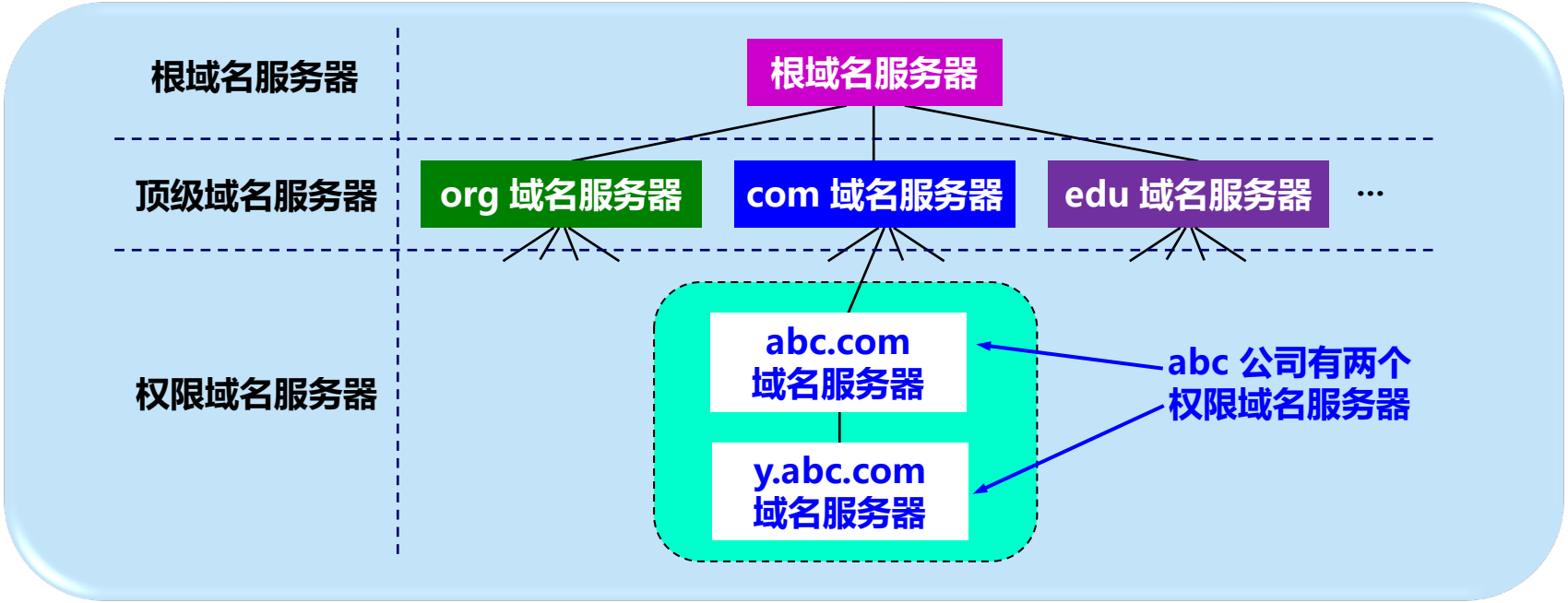
DNS 服务器的管辖范围不是以“域”为单位，而是以“区”为单位。

区的不同划分方法举例



(a) 区 = 域　　(b) 区 < 域

树状结构的 DNS 域名服务器



每个域名服务器都只对域名体系中的一部分进行管辖。

域名服务器类型

- 根据所起的作用，分为四种类型：

1. 根域名服务器
2. 顶级域名服务器
3. 权限域名服务器
4. 本地域名服务器

1. 根域名服务器

- 最高层次，最为重要。
- 所有根域名服务器都知道所有的顶级域名服务器的域名和 IP 地址。
- 不管是哪一个本地域名服务器，若要对互联网上任何一个域名进行解析，只要自己无法解析，就首先求助于根域名服务器。
- 若所有的根域名服务器都瘫痪了，整个互联网中的 DNS 系统就无法工作了。

根域名服务器共有 13 套装置

- 根域名服务器共有 13 套装置，构成 13 组根域名服务器。
- 根域名服务器总共只有 13 个不同 IP 地址的域名，但并非仅由13台机器所组成。

域名	IP 地址	运营商
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	Verisign, Inc.
b.root-servers.net	199.9.14.201, 2001:500:200::b	University of Southern California, Information Sciences Institute
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10, 2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4, 2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53, 2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	Verisign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

根域名服务器共有 13 套装置

- 根域名服务器分布在全世界。
- 为了提供更可靠的服务，在每一个地点的根域名服务器往往由多台机器组成。
- 根域名服务器采用**任播** (anycast) 技术，当DNS 客户向某个根域名服务器发出查询报文时，路由器能找到离这个 DNS 客户**最近**的一个根域名服务器。
- 截至 2020年9月3日，全球共有 1098 个根域名服务器在运行，其中在我国的共有28个。

根域名服务器共有 13 套装置

注意：

根域名服务器并**不直接**把域名转换成 IP 地址（根域名服务器也没有存放这种信息），而是**告诉本地域名服务器**下一步应当找哪一个**顶级域名服务器**进行查询。

2. 顶级域名服务器

- 顶级域名服务器（即 TLD 服务器）负责管理在该顶级域名服务器注册的所有二级域名。
- 当收到 DNS 查询请求时，就给出相应的回答（可能是最后的结果，也可能是下一步应当找的域名服务器的 IP 地址）。

3. 权限域名服务器

- 负责一个区（zone）的域名服务器。
- 当一个权限域名服务器还不能给出最后的查询回答时，就会告诉发出查询请求的 DNS 客户，下一步应当找哪一个权限域名服务器。

4. 本地域名服务器

- **非常重要。**
- 当一个主机发出 DNS 查询请求时，该查询请求报文就**发送**给本地域名服务器。
- 每一个互联网服务提供者 ISP 或一个大学，都可以拥有一个本地域名服务器。
- 当所要查询的主机也属于同一个本地 ISP 时，该本地域名服务器立即就能将所查询的主机名转换为它的 IP 地址，而**不需要**再去询问其他的域名服务器。
- 本地域名服务器有时也称为**默认域名服务器。**

提高域名服务器的可靠性

- DNS 域名服务器都把数据复制到几个域名服务器来保存，其中的一个是**主域名服务器**，其他的是**辅助域名服务器**。
- 当主域名服务器出故障时，辅助域名服务器可以保证 DNS 的查询工作不会中断。
- 主域名服务器**定期**把数据复制到辅助域名服务器中，而**更改**数据**只能**在主域名服务器中进行，保证了数据的一致性。

域名的解析过程

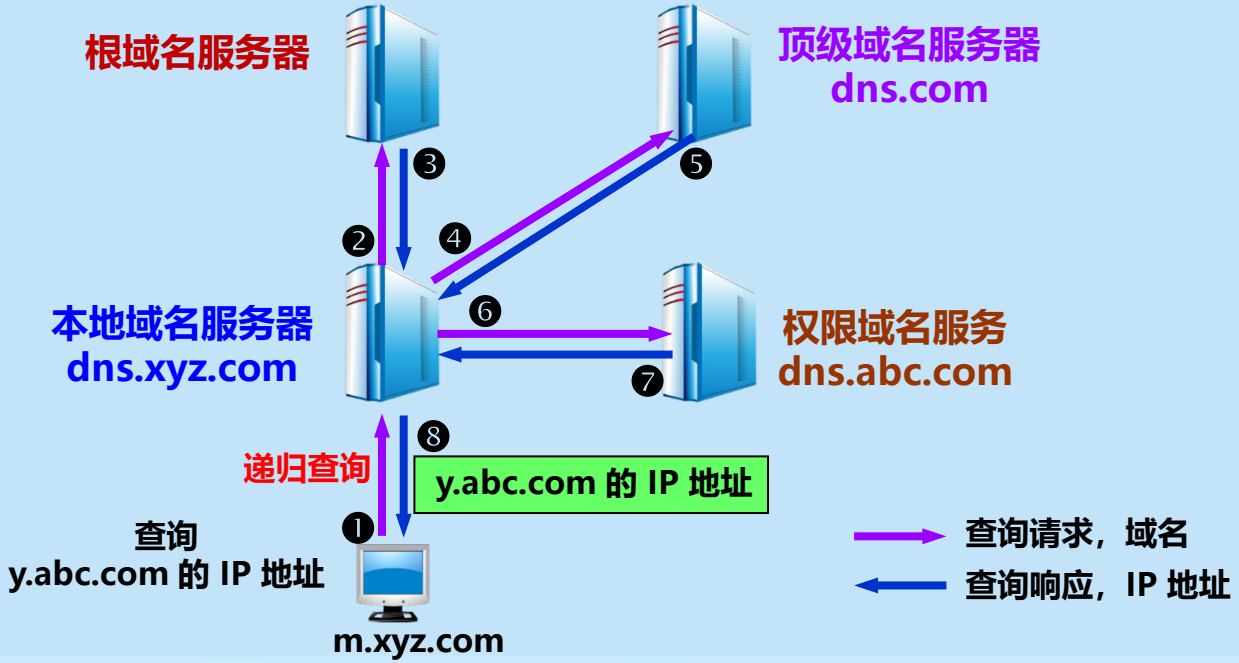
递归查询

- 通常，主机向本地域名服务器查询时使用。
- 若不知道，就以 DNS 客户的身份，向其他根域名服务器继续发出查询请求报文。

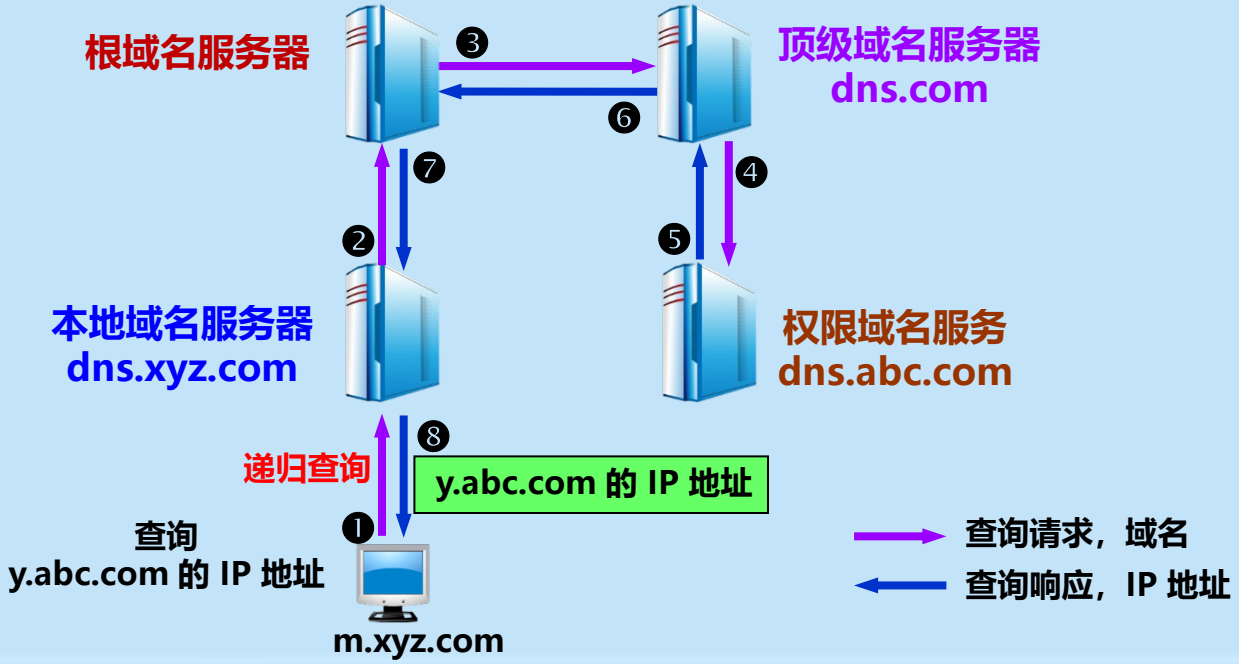
迭代查询

- 本地域名服务器向根域名服务器查询时使用。
- 要么给出所要查询的 IP 地址，要么告诉下一个要查询的域名服务器的 IP 地址。
- 本地域名服务器继续后续查询。

迭代查询


根域名服务器
顶级域名服务器
dns.com
本地域名服务器
dns.xyz.com
权限域名服务
dns.abc.com
递归查询
y.abc.com 的 IP 地址
查询
y.abc.com 的 IP 地址
m.xyz.com
查询请求，域名
查询响应，IP 地址

递归查询（比较少用）


根域名服务器
顶级域名服务器
dns.com
本地域名服务器
dns.xyz.com
权限域名服务
dns.abc.com
递归查询
y.abc.com 的 IP 地址
查询
y.abc.com 的 IP 地址
m.xyz.com
查询请求，域名
查询响应，IP 地址

高速缓存

- 也称为**高速缓存域名服务器。**
- 存放**最近用过的名字**以及从何处获得名字映射信息的记录。
- **作用：**大大减轻根域名服务器的负荷，使 DNS 查询请求和回答报文的数量大为减少。
- 域名服务器应为每项内容设置**计时器，**并处理超过合理时间的项。
- 当权限域名服务器回答一个查询请求时，在响应中指明绑定**有效存在的时间值。**增加此时间值可减少网络开销，而减少此时间值可提高域名转换的准确性。

6.2 文件传送协议

6.2.1 FTP 概述

6.2.2 FTP 的基本工作原理

6.2.3 简单文件传送协议 TFTP

6.2.1 FTP 概述

- 文件传送协议 FTP (File Transfer Protocol) 是互联网上使用得最广泛的文件传送协议。
- 提供交互式的访问，允许客户指明文件的类型与格式，并允许文件具有存取权限。
- 屏蔽了各计算机系统的细节，因而适合于在异构网络中任意计算机之间传送文件。
- 是文件共享协议的一个大类。

文件共享协议

- **文件传送协议：FTP， TFTP 等。**
  - 复制整个文件。对文件副本进行访问。
    - 若要存取一个文件，就必须先获得一个本地文件副本。
    - 若要修改文件，只能对文件副本进行修改，然后再将修改后的文件副本传回到原节点。
- **联机访问 (on-line access) 协议：NFS 等。**
  - 允许同时对一个文件进行存取。
  - 远地共享文件访问，如同对本地文件的访问一样。
  - 透明存取，不需要对该应用程序作明显的改动。
  - **由操作系统负责。**

6.2.2 FTP 的基本工作原理

网络环境下复制文件的**复杂性**：

- 计算机存储数据的格式不同。
- 文件的目录结构和文件命名的规定不同。
- 对于相同的文件存取功能，操作系统使用的命令不同。
- 访问控制方法不同。

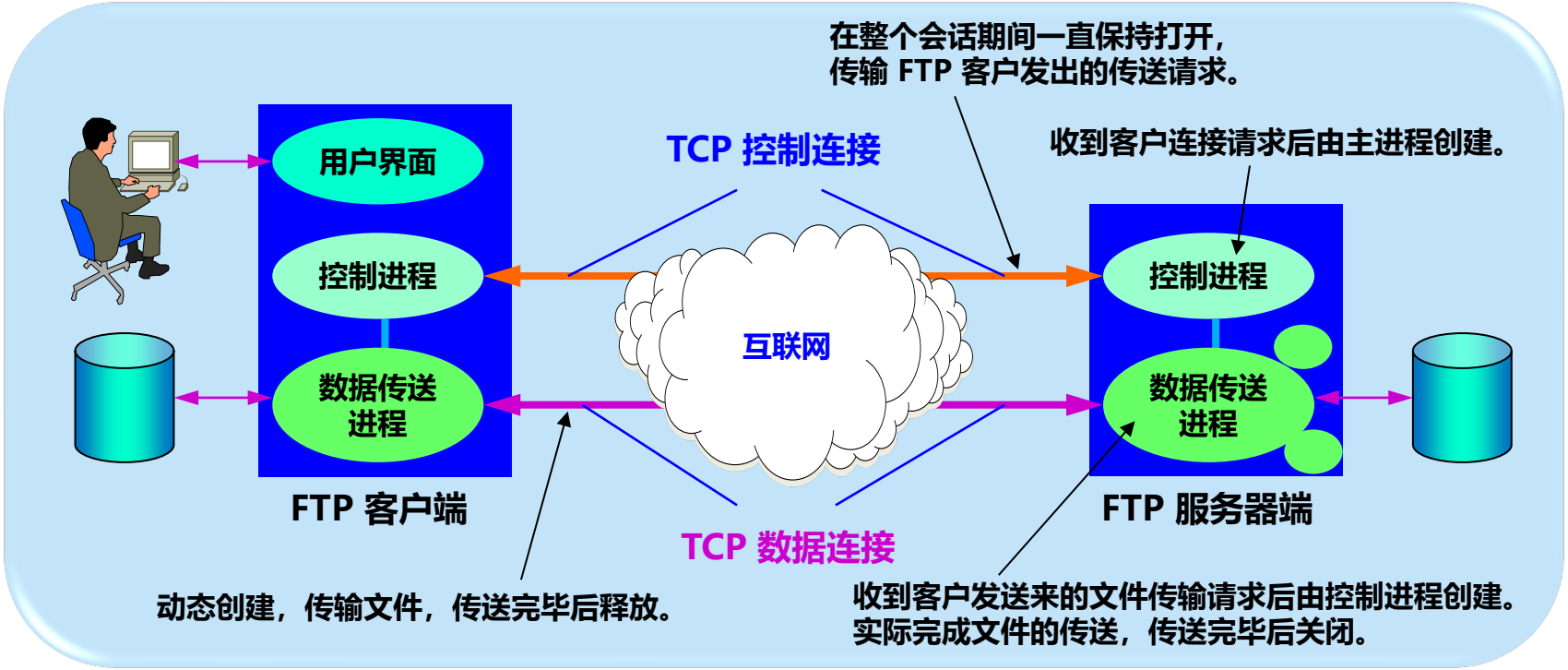
FTP 特点

- 只提供文件传送的一些基本服务，它使用 TCP 可靠的运输服务。
- 主要功能：减少或消除在不同操作系统下处理文件的不兼容性。
- 使用客户服务器方式。
  - 一个 FTP 服务器进程可同时为多个客户进程提供服务。
  - FTP 的服务器进程由两大部分组成：
    - 一个主进程，负责接受新的请求；
    - 若干个从属进程，负责处理单个请求。

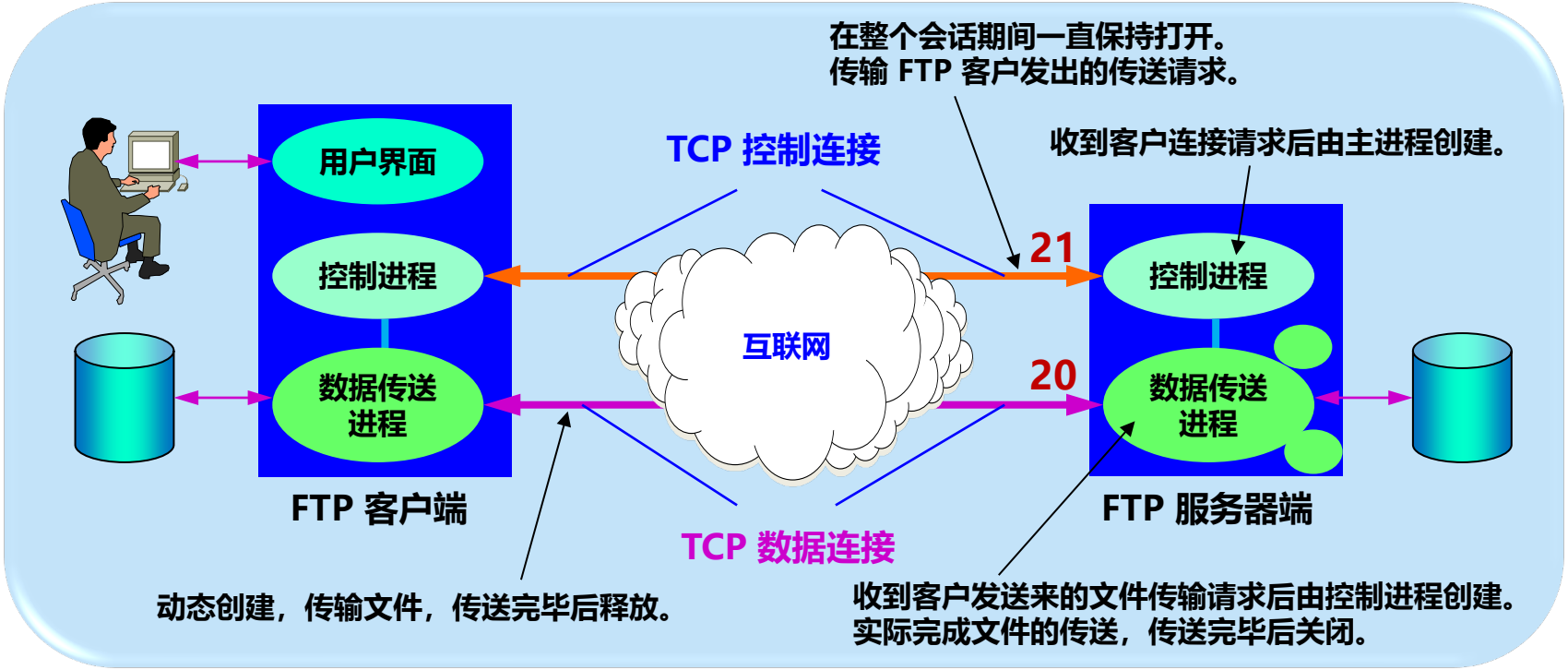
FTP 主进程的工作步骤

1. 打开熟知端口（端口号为 21），使客户进程能够连接上。
2. 等待客户进程发出连接请求。
3. 启动从属进程来处理客户进程发来的请求。从属进程对客户进程的请求处理完毕后即终止，但从属进程在运行期间根据需要还可能创建其他一些子进程。
4. 回到等待状态，继续接受其他客户进程发来的请求。主进程与从属进程的处理是并发地进行。

FTP 客户和服务器之间的两个从属进程和两个 TCP 连接



FTP 使用两个不同的端口号



NFS 采用的思路

- FTP 并非对所有的数据传输都是最佳的：仅能访问副本。
- NFS 允许应用进程打开一个远地文件，并能在该文件的某一个特定的位置上开始读写数据。
- NFS 可使用户只复制一个大文件中的一个很小的片段，而不需要复制整个大文件。
- NFS 在网络上传送的只是少量的修改数据。

6.2.3 简单文件传送协议 TFTP

- TFTP (Trivial File Transfer Protocol) 是一个很小且易于实现的文件传送协议。
- 使用客户服务器方式和使用 UDP 数据报，因此 TFTP 需要有自己的差错改正措施。
- 只支持文件传输，不支持交互。
- 没有庞大的命令集，没有列目录的功能，也不能对用户进行身份鉴别。
- 优点：（1）可用于 UDP 环境；（2）代码所占的内存较小。

TFTP 的主要特点

- 每次传送的数据报文中有 512 字节的数据，但最后一次可不足 512 字节。
- 数据报文按序编号，从 1 开始。
- 支持 ASCII 码或二进制传送。
- 可对文件进行读或写。
- 使用很简单的首部。

TFTP 的工作很像停止等待协议

- 发送完一个文件块后就等待对方的确认，确认时应指明所确认的块编号。
- 发完数据后在规定时间内收不到确认就要重发数据 PDU。
- 发送确认 PDU 的一方若在规定时间内未收到下一个文件块，需重发确认 PDU，保证文件的传送不致因某一个数据报的丢失而告失败。

TFTP 的工作过程

- 开始工作时，TFTP 客户进程发送一个读请求或写请求报文给 TFTP 服务器进程，其 UDP 熟知端口号码为 69。
- TFTP 服务器进程选择一个新的端口和 TFTP 客户进程进行通信。
- 若文件长度恰好为 512 字节的整数倍，则在文件传送完毕后，还必须在最后发送一个只含首部而无数据的数据报文。
- 若文件长度不是 512 字节的整数倍，则最后传送数据报文的数据字段一定不满 512 字节，作为文件结束的标志。

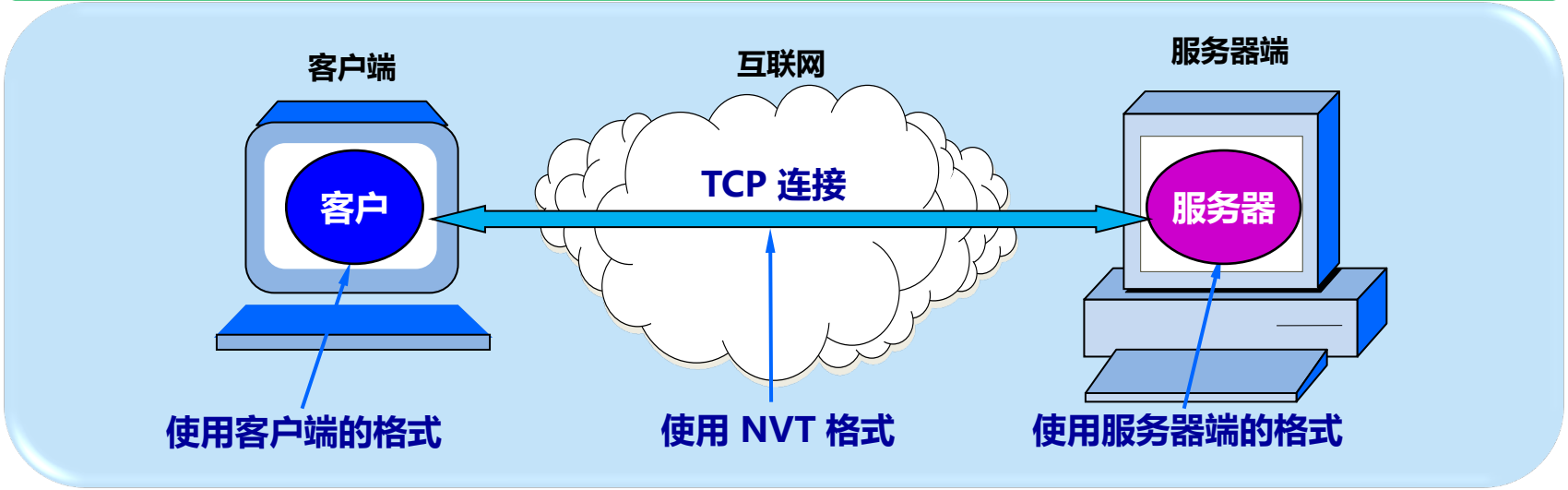
6.3 远程终端协议 TELNET

- 是一个简单的远程终端协议，是互联网的正式标准。
- 允许用户在其所在地通过 TCP 连接注册（即登录）到远地的另一个主机上（使用主机名或 IP 地址）。
- 能将用户的击键传到远地主机，同时也能将远地主机的输出通过 TCP 连接返回到用户屏幕。
- 服务是透明的。
- 又称为终端仿真协议。

TELNET 使用客户 - 服务器方式

- 在本地系统运行 TELNET **客户进程**，而在远地主机则运行 TELNET **服务器进程**。
- 服务器中的**主进程**等待新的请求，产生**从属进程**来处理每一个连接。

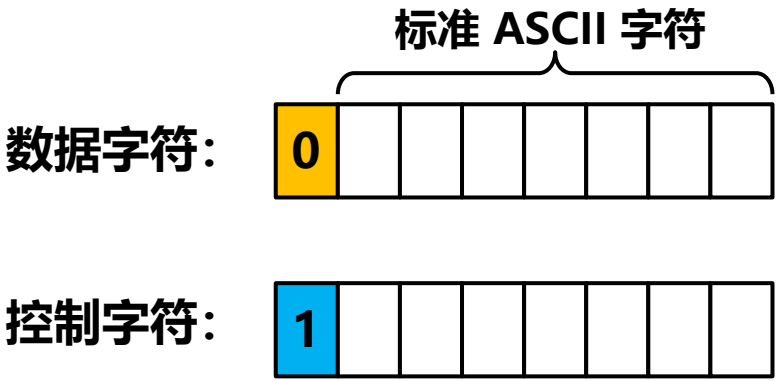
TELNET 使用网络虚拟终端 NVT 格式



TELNET 的选项协商 (Option Negotiation) 使客户和服务器可商定使用更多的终端功能，协商的双方是平等的。

NVT （Network Virtual Terminal ）格式

两个字符集：数据，控制



- 客户端把用户的击键和命令转换成 NVT 格式，并送交服务器。
- 服务器端把收到的数据和命令，从 NVT 格式转换成远地系统所需的格式。
- 向客户返回数据时，服务器把远地系统的格式转换为 NVT 格式，本地客户再从 NVT 格式转换到本地系统所需的格式。

6.4 万维网 WWW

6.4.1 万维网概述

6.4.2 统一资源定位符 URL

6.4.3 超文本传送协议 HTTP

6.4.4 万维网的文档

6.4.5 万维网的信息检索系统

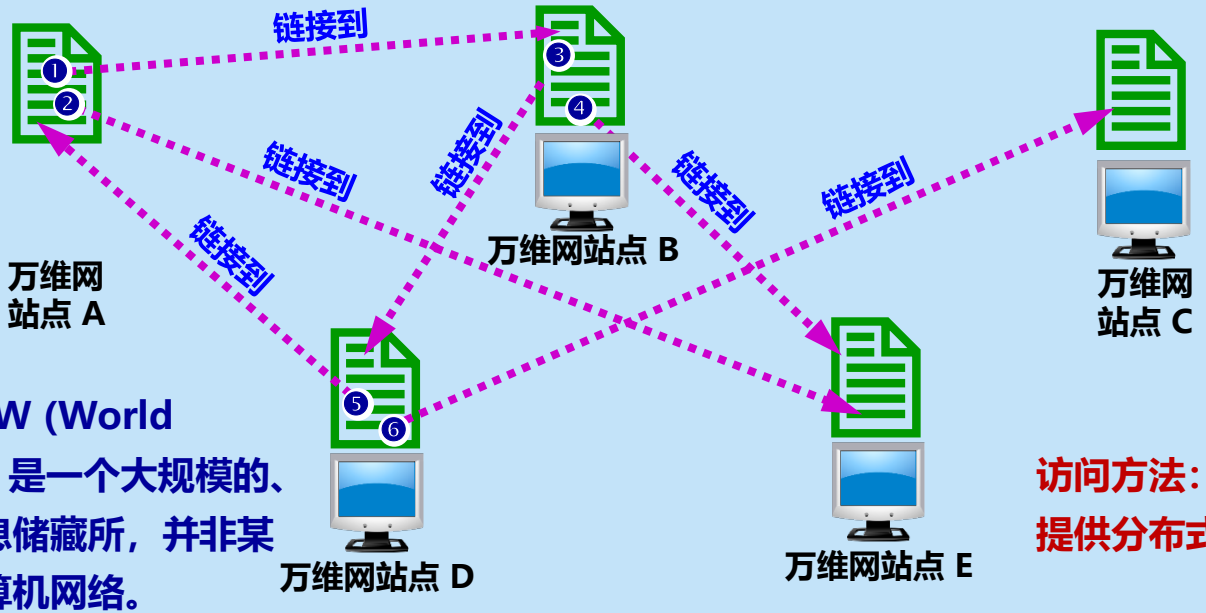
6.4.6 博客和微博

6.4.7 社交网站

6.4.1 万维网概述

- **万维网 WWW (World Wide Web) 并非某种特殊的计算机网络。**
- **万维网是一个大规模的、联机式的信息储藏所。**
- **万维网用链接的方法能非常方便地从互联网上的一个站点访问另一个站点，从而主动地按需获取丰富的信息。**
- **这种访问方式称为“链接”。**

6.4.1 万维网概述



万维网 WWW (World Wide Web) 是一个大规模的、联机式的信息储藏所，并非某种特殊的计算机网络。

访问方法：链接。
提供分布式服务。

万维网是分布式超媒体 (hypermedia) 系统

- 是超文本 (hypertext) 系统的扩充。
- 超文本：包含指向其他文档的链接的文本，是万维网的基础。
- 超媒体与超文本的区别：文档内容不同。
  - 超文本文档仅包含文本信息。
  - 超媒体文档还包含其他信息，如图形、图像、声音、动画，甚至活动视频图像等。
- 分布式系统
  - 信息分布在整个互联网上。每台主机上的文档都独立进行管理。

万维网的工作方式

- 以客户服务器方式工作。
- 客户程序：浏览器。
- 服务器程序：在万维网文档所驻留的主机上运行。这个计算机也称为万维网服务器。
- 客户程序向服务器程序发出请求，服务器程序向客户程序送回客户所要的万维网文档。
- 在一个客户程序主窗口上显示出的万维网文档称为页面 (page)。

万维网必须解决的问题（1/2）

(1) 怎样标志分布在整个互联网上的万维网文档?

- 使用统一资源定位符 URL (Uniform Resource Locator) 。
- 使每一个文档在整个互联网的范围内具有唯一的标识符 URL。

(2) 用什么协议来实现万维网上的各种链接?

- 使用超文本传送协议 HTTP (HyperText Transfer Protocol)。
- HTTP 是一个应用层协议，使用 TCP 连接进行可靠的传送。

万维网必须解决的问题（2/2）

(3) 怎样使不同作者创作的不同风格的万维网文档都能在互联网上的各种主机上显示出来，同时使用户清楚地知道在什么地方存在着链接?

- 使用超文本标记语言 HTML (HyperText Markup Language)。

(4) 怎样使用户能够很方便地找到所需的信息?

- 使用各种的搜索工具（即搜索引擎）。

6.4.2 统一资源定位符 URL

- 是对互联网上资源的位置和访问方法的一种**简洁表示。**
- 给资源的位置提供一种**抽象**的识别方法，并用这种方法给**资源定位。**
- 实际上就是在互联网上的**资源的地址。**
- 显然，互联网上的所有资源，都有一个**唯一**确定的URL。
- **资源：**指在互联网上可以被访问的任何对象，包括文件目录、文件、文档、图像、声音等，以及与互联网相连的**任何形式**的数据。

URL 相当于一个文件名在网络范围的扩展。因此，URL 是与互联网相连的机器上的任何可访问对象的一个指针。

1. URL 的格式

- 由以冒号（:）隔开的两大部分组成，对字符大写或小写没有要求。
- 一般形式：

<协议>://<主机>:<端口>/<路径>

- ftp —— 文件传送协议 FTP
- http —— 超文本传送协议 HTTP
- News —— USENET 新闻

1. URL 的格式

- 由以冒号（:）隔开的两大部分组成，对字符大写或小写没有要求。
- 一般形式：

<协议>://<主机>:<端口>/<路径>

规定的格式

1. URL 的格式

- 由以冒号（:）隔开的两大部分组成，对字符大写或小写没有要求。
- 一般形式：

<协议>://<主机>:<端口>/<路径>

存放资源的主机在互联网中的域名，也可以是用点分十进制的 IP 地址。

1. URL 的格式

- 由以冒号（:）隔开的两大部分组成，对字符大写或小写没有要求。
- 一般形式：

<协议>://<主机>:<端口>/<路径>

端口号。省略时
使用默认端口号。

1. URL 的格式

- 由以冒号（:）隔开的两大部分组成，对字符大写或小写没有要求。
- 一般形式：

<协议>://<主机>:<端口>/<路径>

资源所在目录位置。区分大小写。省略时使用所定义的默认路径。后面可能还有一些选项。

2. 使用 HTTP 的 URL

使用 HTTP 协议。

规定的格式。

主机域名或 IP 地址。

HTTP 的默认端口号是 80，可省略。

http://<主机>:<端口>/<路径>

省略时指到互联网上的某个主页 (home page)。更复杂一些的路径是指向层次结构的从属页面。

http://www.tsinghua.edu.cn/publish/newthu/newthu_cnt/faculties/index.html

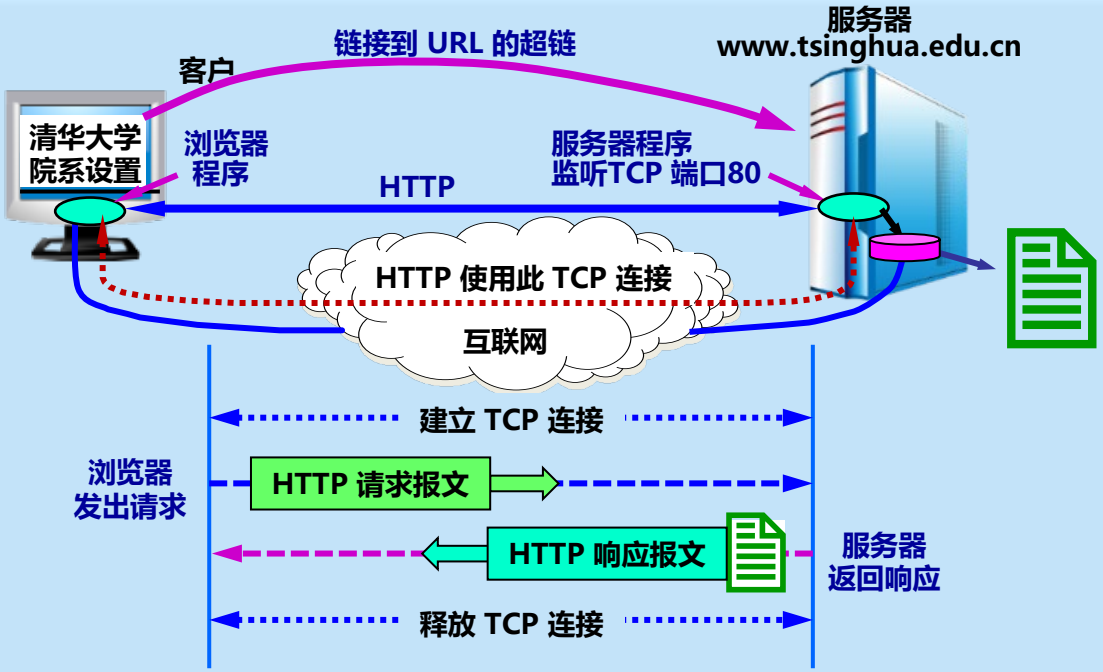
主机域名　　路径名

6.4.3 超文本传送协议 HTTP

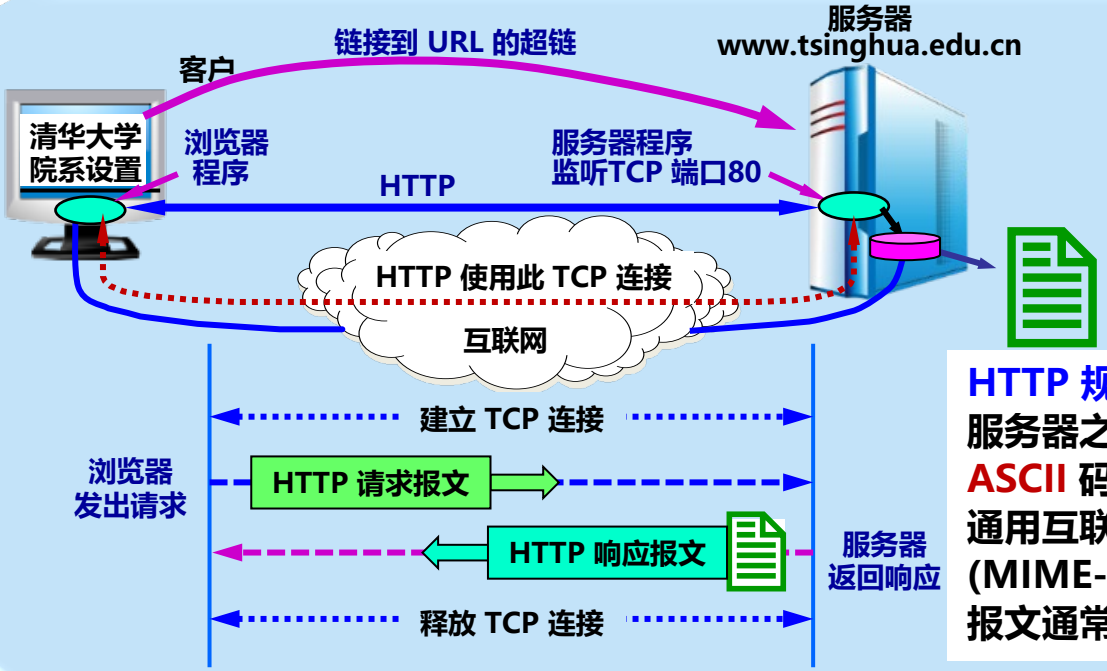
- HTTP 是面向事务的 (transaction-oriented) 应用层协议。
- 使用 TCP 连接进行可靠的传送。
- 定义了浏览器与万维网服务器通信的格式和规则。
- 是万维网上能够可靠地交换文件（包括文本、声音、图像等各种多媒体文件）的重要基础。

HTTP 不仅传送完成超文本跳转所必需的信息，而且也传送任何可从互联网上得到的信息，如文本、超文本、声音和图像等。

1. HTTP 的操作过程


服务器
www.tsinghua.edu.cn
链接到 URL 的超链
客户
清华大学
院系设置
浏览器
程序
服务器程序
监听TCP 端口80
HTTP
HTTP 使用此 TCP 连接
互联网
建立 TCP 连接
浏览器
发出请求
HTTP 请求报文
HTTP 响应报文
服务器
返回响应
释放 TCP 连接

1. HTTP 的操作过程



HTTP 规定：在 HTTP 客户与 HTTP 服务器之间的每次交互，都由一个 ASCII 码串构成的请求和一个类似的通用互联网扩充，即“类MIME (MIME-like)”的响应组成。HTTP 报文通常都使用 TCP 连接传送。

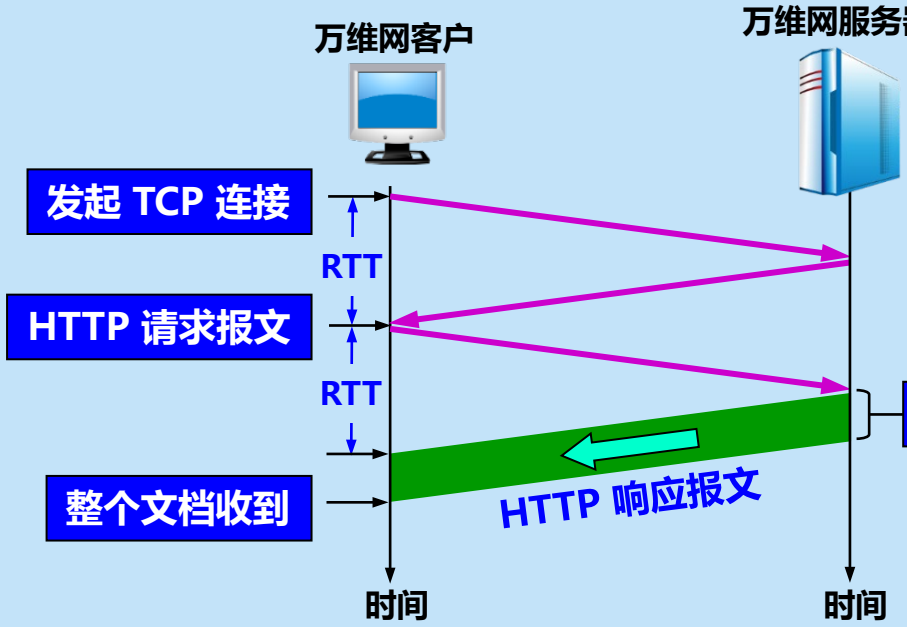
用户浏览页面的两种方法

1. 在浏览器的地址窗口中键入所要找的页面的 URL。
2. 在某一个页面中用鼠标点击一个可选部分，这时浏览器会自动在互联网上找到所要链接的页面。

HTTP 的主要特点

- HTTP 使用了面向连接的 TCP 作为运输层协议，保证了数据的可靠传输。
- HTTP 协议本身也是**无连接的。**
- HTTP 是**无状态的** (stateless)，简化了服务器的设计，使服务器更容易支持大量并发的 HTTP 请求。

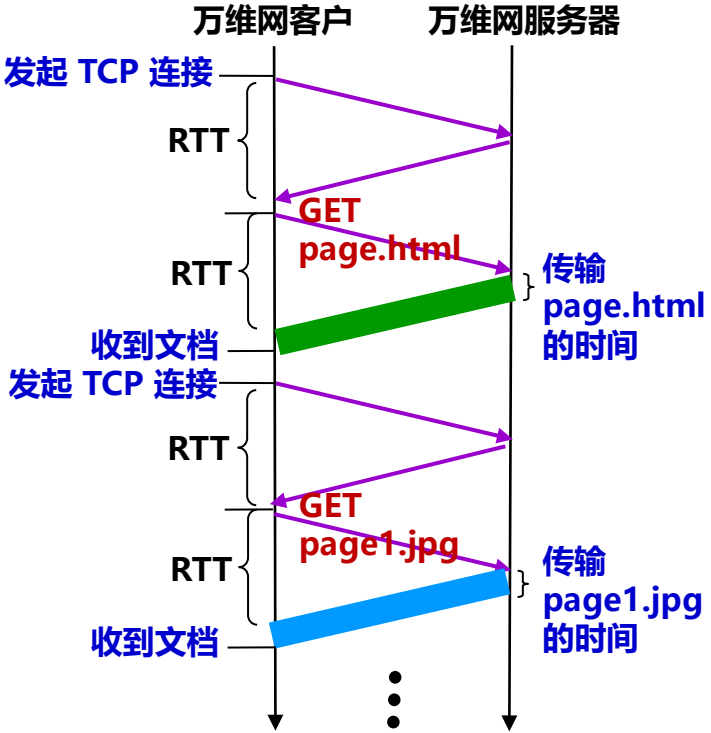
请求一个万维网文档所需的时间



所需的时间 >=
RTT（三报文握手建立 TCP 连接）
+ RTT（请求和接收文档）
+ 文档的传输时间
= 2 RTT +文档的传输时间

协议 HTTP/1.0 的主要缺点

- 每请求一个文档就要有两倍 RTT 的开销。
- 客户和服务器每一次建立新的 TCP 连接都要分配缓存和变量。
- 这种**非持续连接**使服务器的负担很重。



假设一个主页 page.html 上有 10 个链接的图片：
page1.jpg,
page2.jpg,
page3.jpg,
…

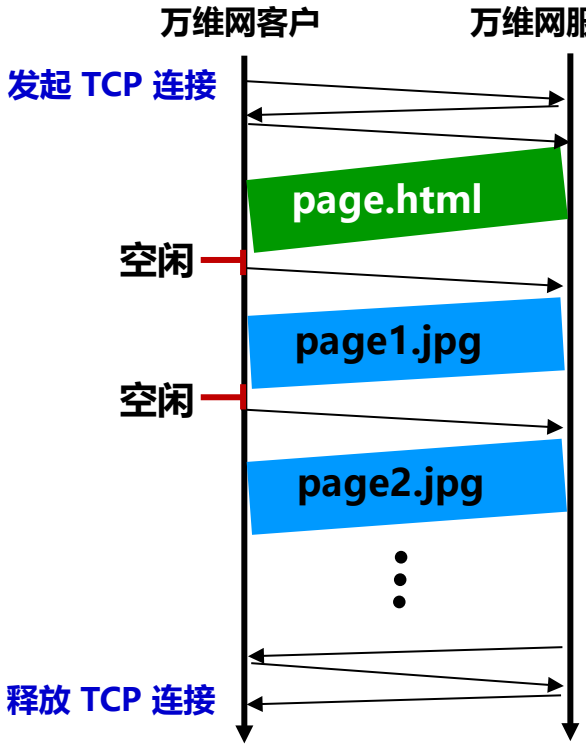
所需的时间 >=
22 RTT
+ 11 个文档的传输时间

协议 HTTP/1.1 使用持续连接

- **持续连接**（persistent connection）：服务器在发送响应后仍然在一段时间内保持这条连接（不释放），使同一个客户（浏览器）和该服务器可以继续在这条连接上传送后续的 HTTP 请求报文和响应报文。
- 只要文档都在同一个服务器上，就可以继续使用该 TCP 连接。
- 两种工作方式：
  - 非流水线方式 (without pipelining)
  - 流水线方式 (with pipelining)。

持续连接：非流水线方式

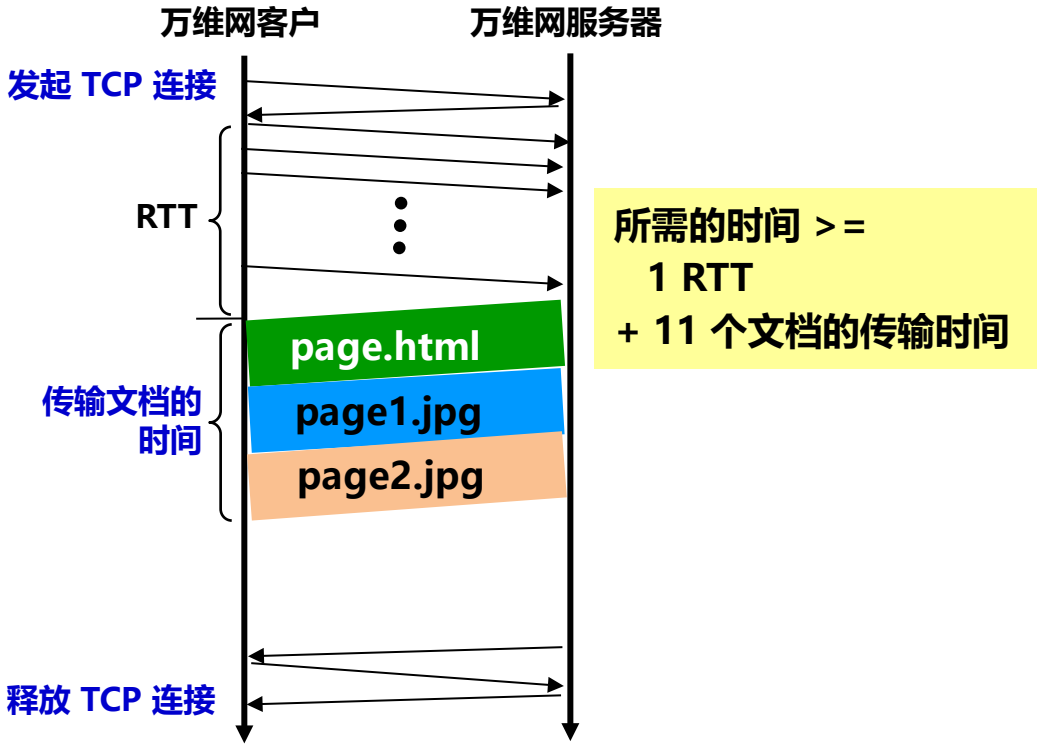
- 客户在收到前一个响应**之后**才能发出下一个请求。
- **缺点：** TCP 连接空闲状态。



所需的时间 >=
2 RTT
+ page.html 传输时间
+ 10 RTT
+ 10 个文档的传输时间
+ 空闲时间

持续连接：流水线方式

- 客户在收到响应报文之前就能够接着发送新的请求报文。
- 连续的多个请求报文到达服务器后，服务器就可连续发回响应报文。
- 下载效率提高。



协议 HTTP/2

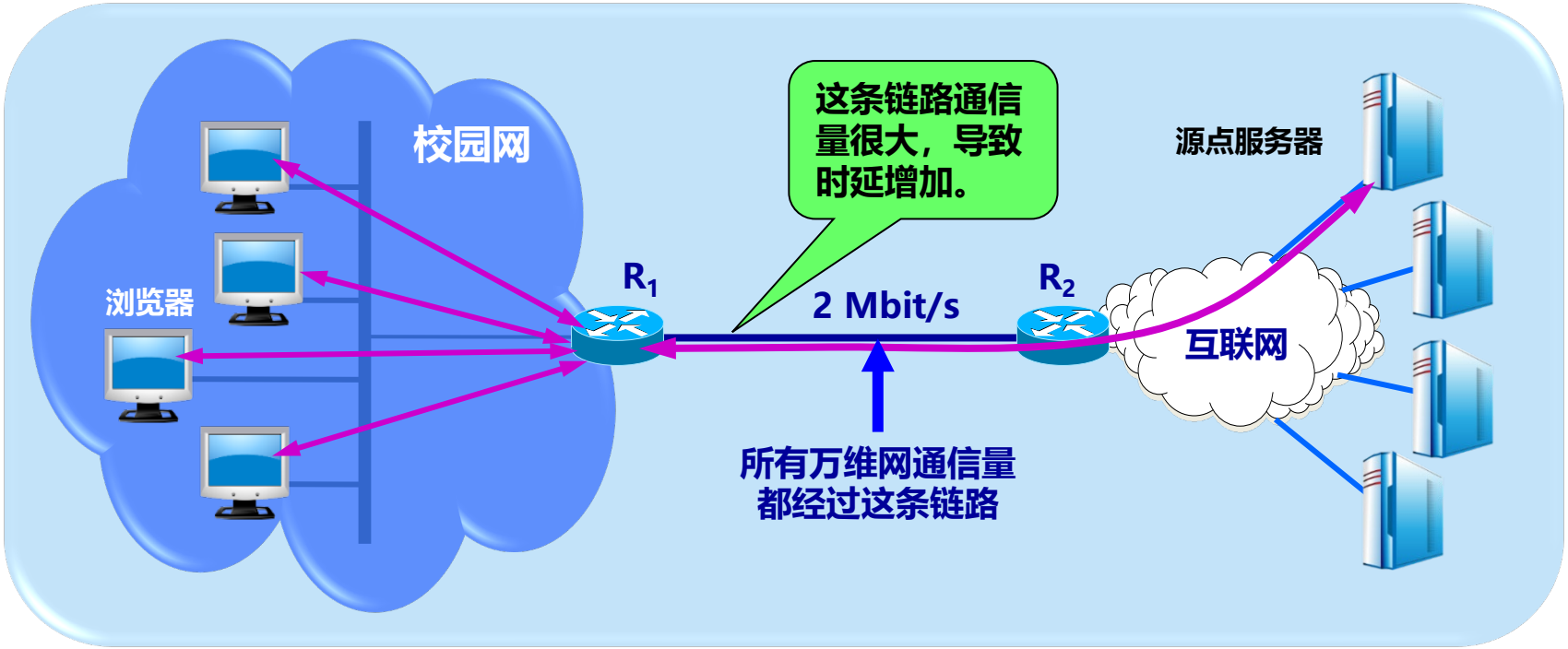
- 是协议 HTTP/1.1 的升级版本。

1. 服务器可以并行发回响应（使用同一个 TCP 连接）。
2. 允许客户复用 TCP 连接进行多个请求。
3. 把所有的报文都划分为许多较小的二进制编码的帧，并采用了新的压缩算法，不发送重复的首部字段，大大减小了首部的开销，提高了传输效率。
4. 向后兼容。

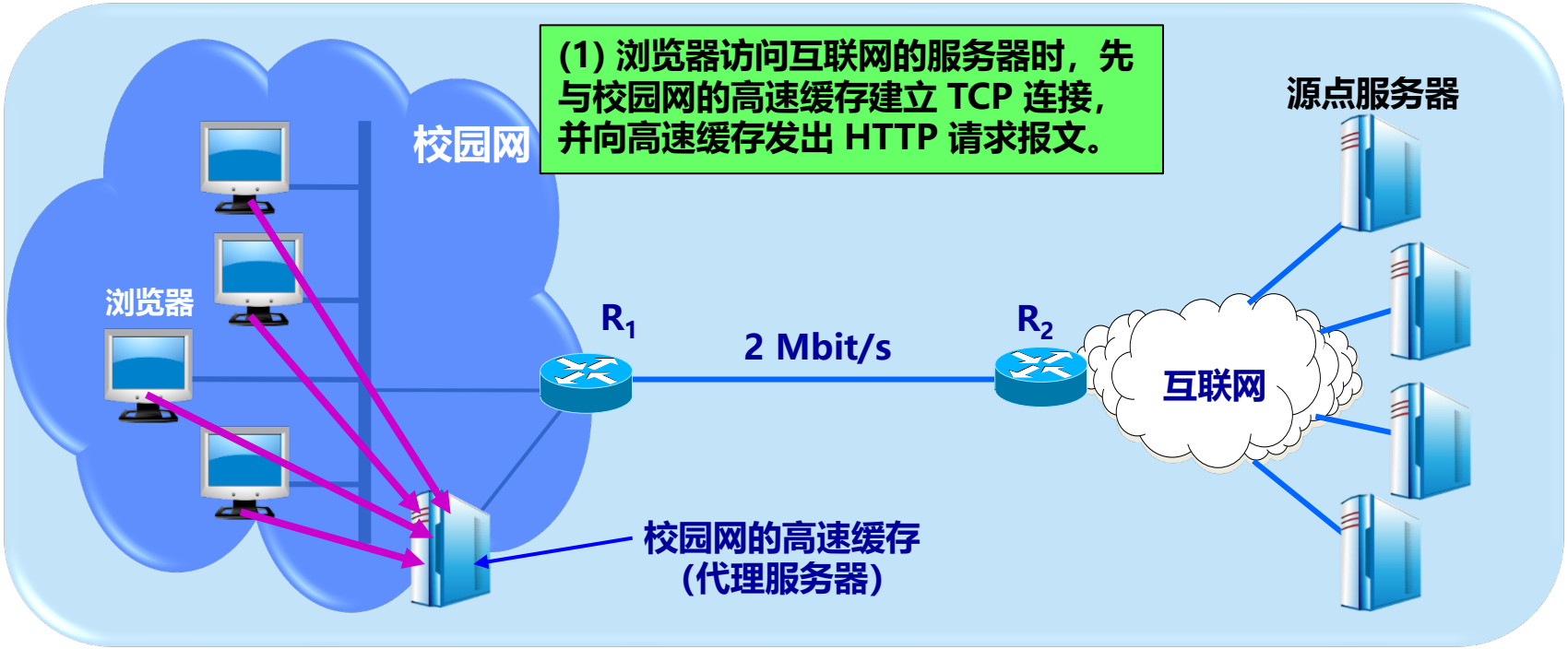
2. 代理服务器

- **代理服务器** (proxy server) 又称为**万维网高速缓存** (Web cache)，它代表浏览器发出 HTTP 请求。
- 使用高速缓存可**减少**访问互联网服务器的**时延。**

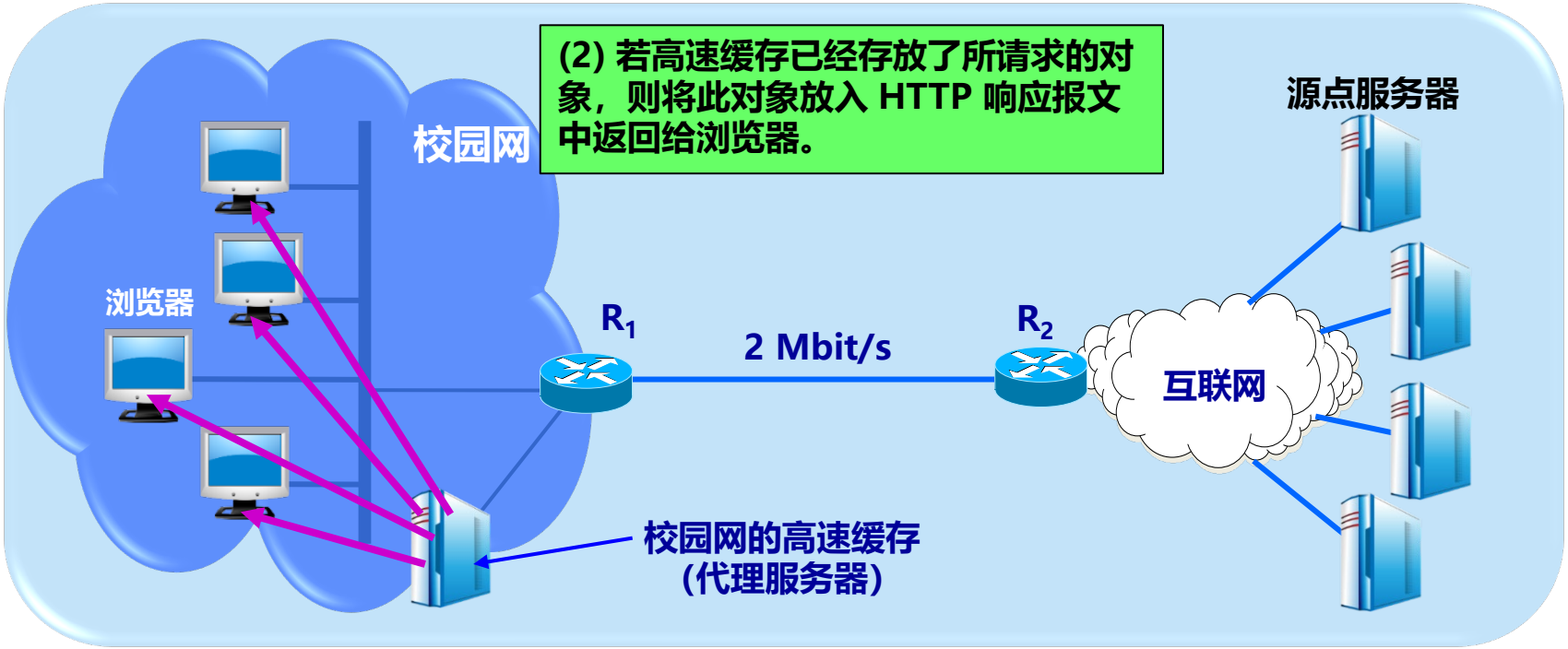
不使用高速缓存的情况


校园网
浏览器
这条链路通信量很大，导致时延增加。
R1
R2
2 Mbit/s
所有万维网通信量都经过这条链路
源点服务器
互联网

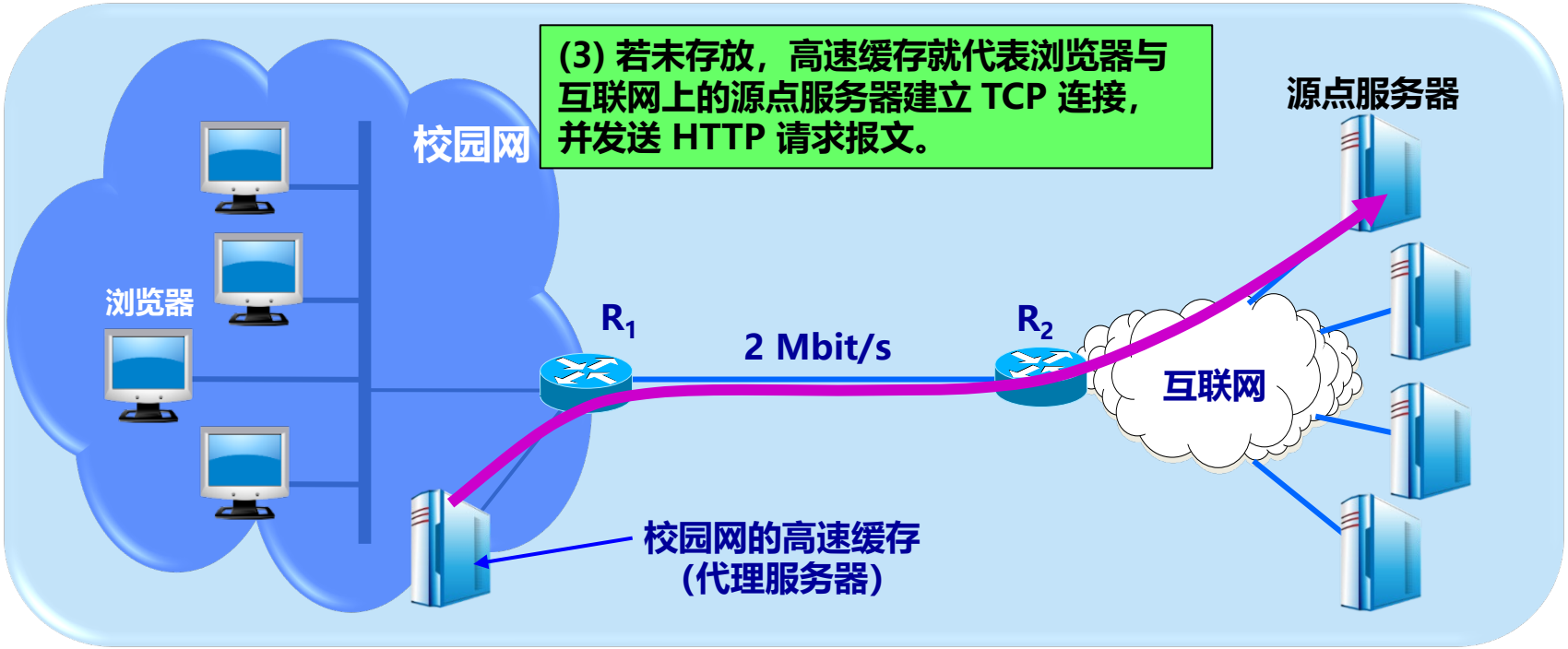
使用高速缓存的情况


(1) 浏览器访问互联网的服务器时，先与校园网的高速缓存建立 TCP 连接，并向高速缓存发出 HTTP 请求报文。
校园网
浏览器
R1
2 Mbit/s
R2
互联网
源点服务器
校园网的高速缓存
(代理服务器)

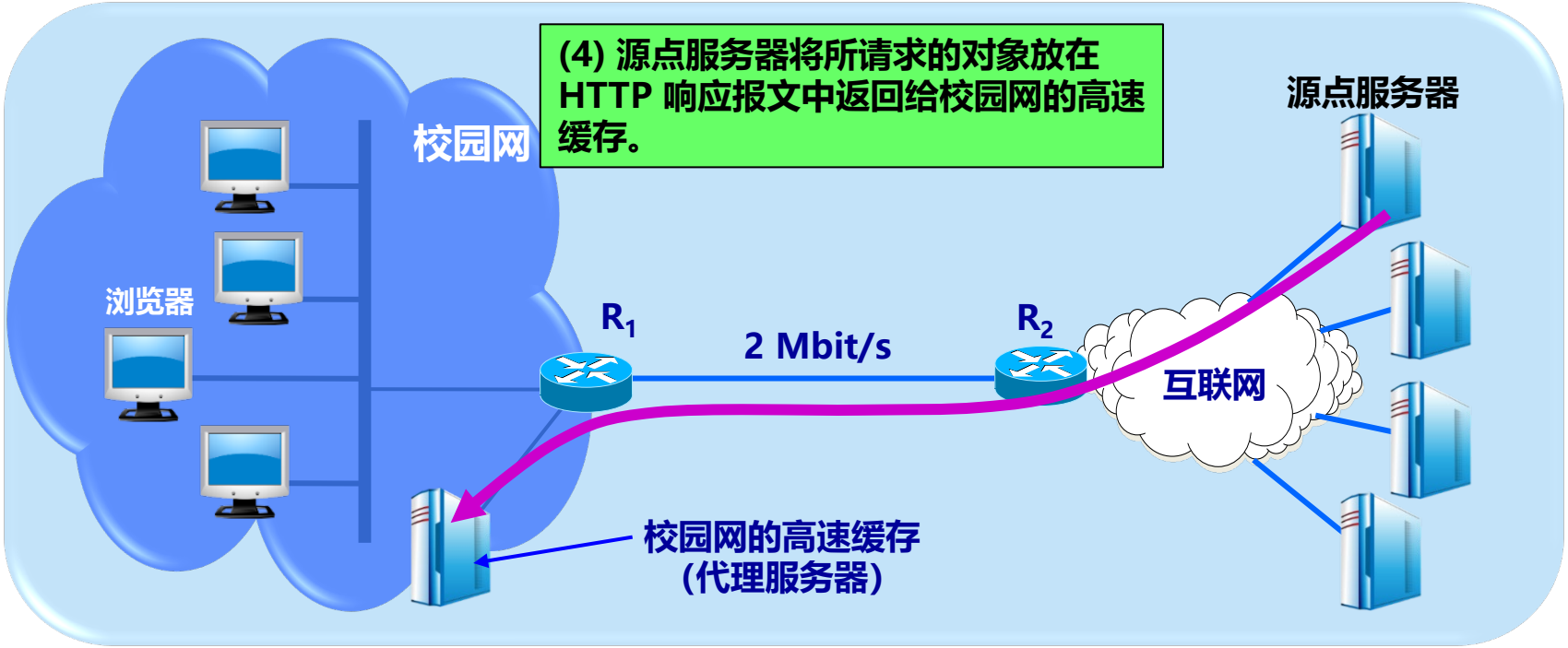
使用高速缓存的情况


(2) 若高速缓存已经存放了所请求的对象，则将此对象放入 HTTP 响应报文中返回给浏览器。
校园网
浏览器
R1
2 Mbit/s
R2
互联网
源点服务器
校园网的高速缓存
(代理服务器)

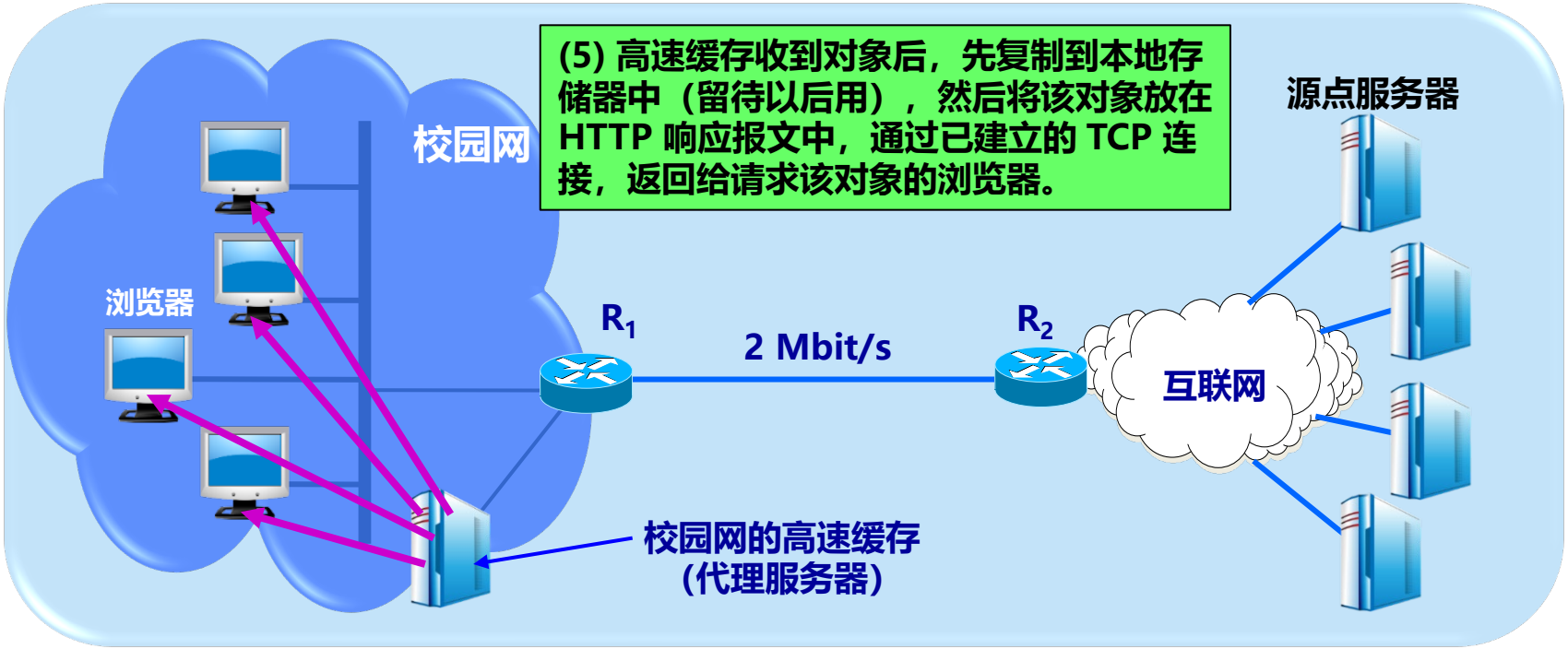
使用高速缓存的情况


(3) 若未存放，高速缓存就代表浏览器与互联网上的源点服务器建立 TCP 连接，并发送 HTTP 请求报文。
校园网
浏览器
R1
2 Mbit/s
R2
互联网
源点服务器
校园网的高速缓存
(代理服务器)

使用高速缓存的情况


(4) 源点服务器将所请求的对象放在 HTTP 响应报文中返回给校园网的高速缓存。
校园网
浏览器
R1
2 Mbit/s
R2
互联网
源点服务器
校园网的高速缓存
(代理服务器)

使用高速缓存的情况


(5) 高速缓存收到对象后，先复制到本地存储器中（留待以后用），然后将该对象放在 HTTP 响应报文中，通过已建立的 TCP 连接，返回给请求该对象的浏览器。
校园网
浏览器
R1
2 Mbit/s
R2
互联网
源点服务器
校园网的高速缓存
（代理服务器）

3. HTTP 的报文结构

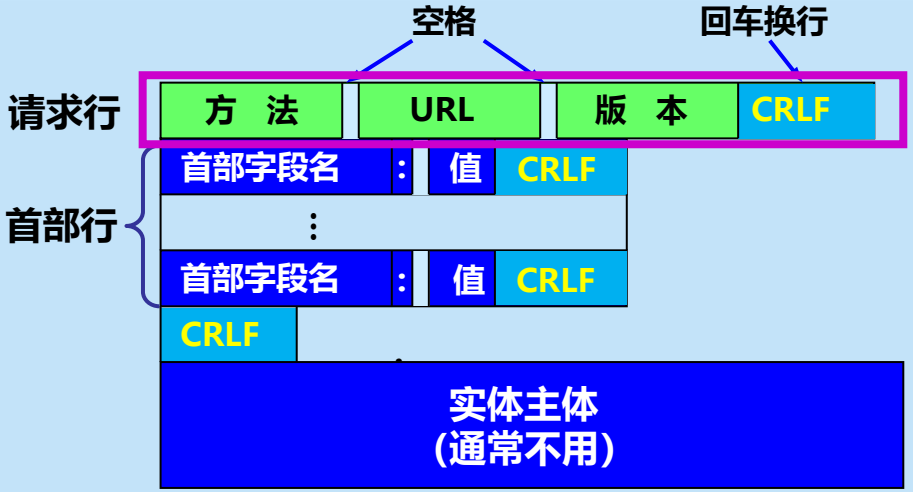
两类报文：

- **请求报文：从客户向服务器的请求。**
- **响应报文：从服务器到客户的回答。**
- **由于 HTTP 是面向正文的 (text-oriented)，因此报文中每一个字段的值都是一些 ASCII 码串，每个字段的长度都是不确定的。**

三个组成部分：

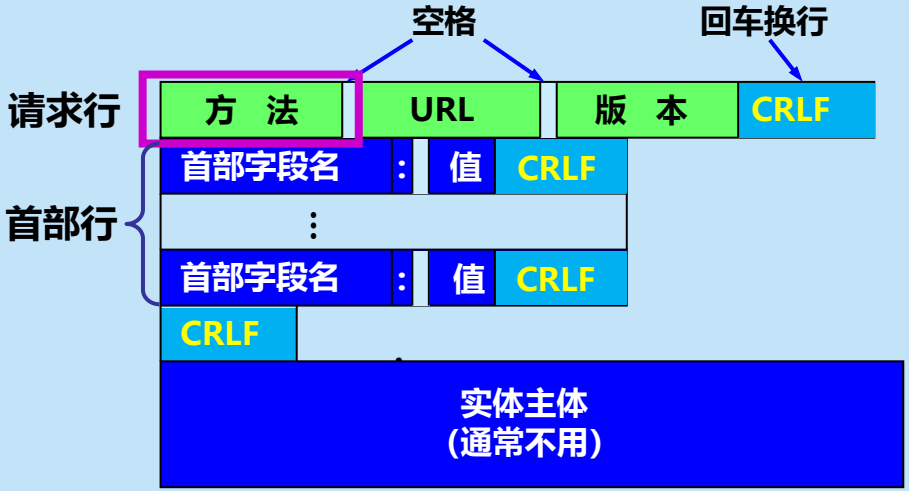
- **开始行：用于区分是请求报文还是响应报文。**
- **首部行：说明浏览器、服务器或报文主体的一些信息。可以有多行，也可以不使用。**
- **实体主体：请求报文中一般不用，响应报文中也可能没有该字段。**

HTTP 的报文结构（请求报文）



在请求报文中，开始行是请求行。

HTTP 的报文结构（请求报文）

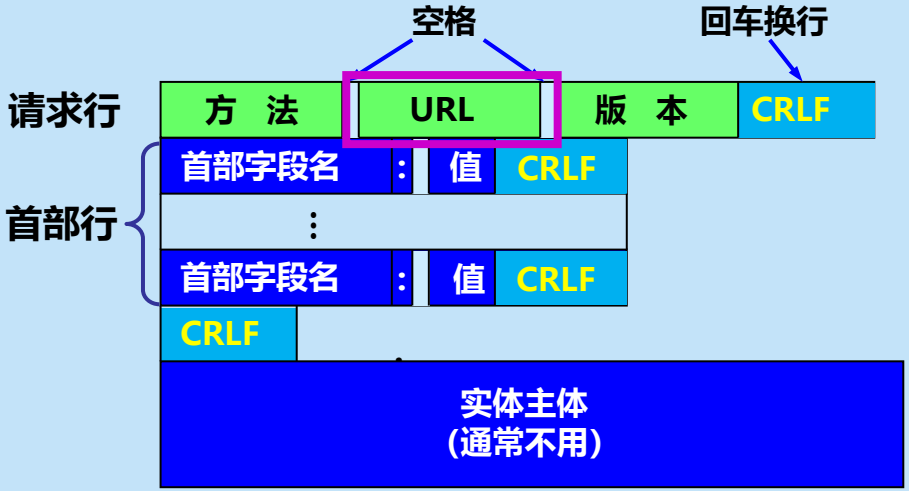


方法：对所请求的对象进行的操作，实际上就是一些命令。
请求报文的类型是由它所采用的方法决定的。

HTTP 请求报文的一些方法

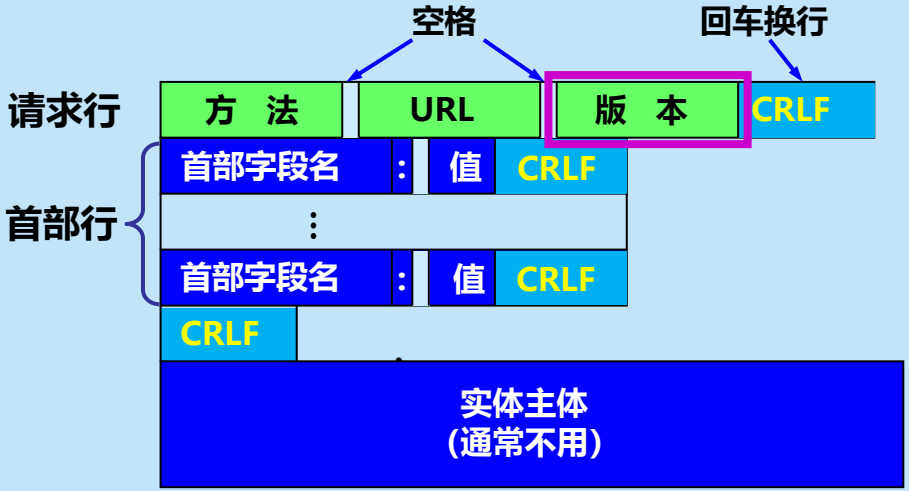
方法（操作）	意义
OPTION	请求一些选项的信息
GET	请求读取由 URL所标志的信息
HEAD	请求读取由 URL所标志的信息的首部
POST	给服务器添加信息（例如，注释）
PUT	在指明的 URL下存储一个文档
DELETE	删除指明的 URL所标志的资源
TRACE	用来进行环回测试的请求报文
CONNECT	用于代理服务器

HTTP 的报文结构（请求报文）



URL：所请求的资源的 URL。

HTTP 的报文结构（请求报文）

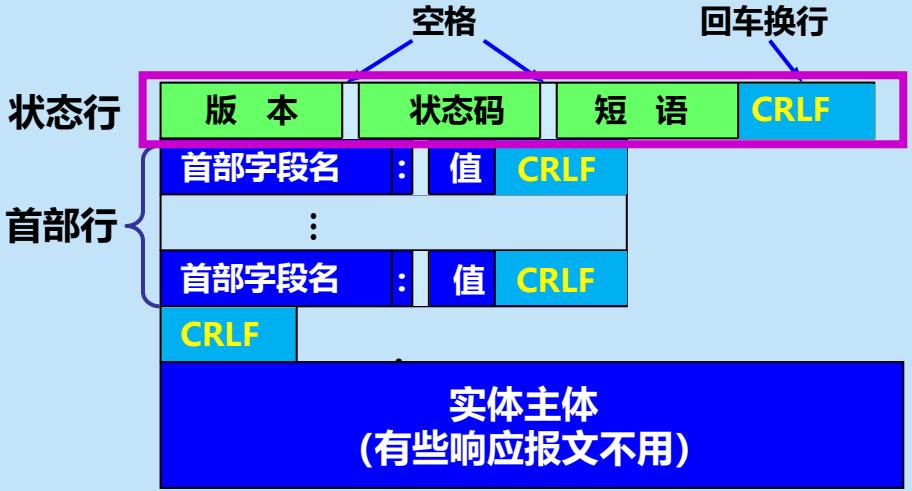


版本：HTTP 的版本。

HTTP 请求报文举例

请求行	GET /dir/index.htm HTTP/1.1	使用了相对 URL
首部行	Host: www.xyz.edu.cn	首部行开始。给出主机的域名
	Connection: close	告诉服务器发送完请求的文档后释放连接
	User-Agent: Mozilla/5.0	表明用户代理使用火狐浏览器 Firefox
	Accept-Language: cn	表示用户希望优先得到中文版本的文档
	CRLF	请求报文的最后一行为空行
		没有实体主体

HTTP 的报文结构（响应报文）



在响应报文中，开始行是状态行。
版本：HTTP 的版本。状态码：服务器操作完成的状态。短语：解释状态码。

状态码：都是三位数字，分为5大类

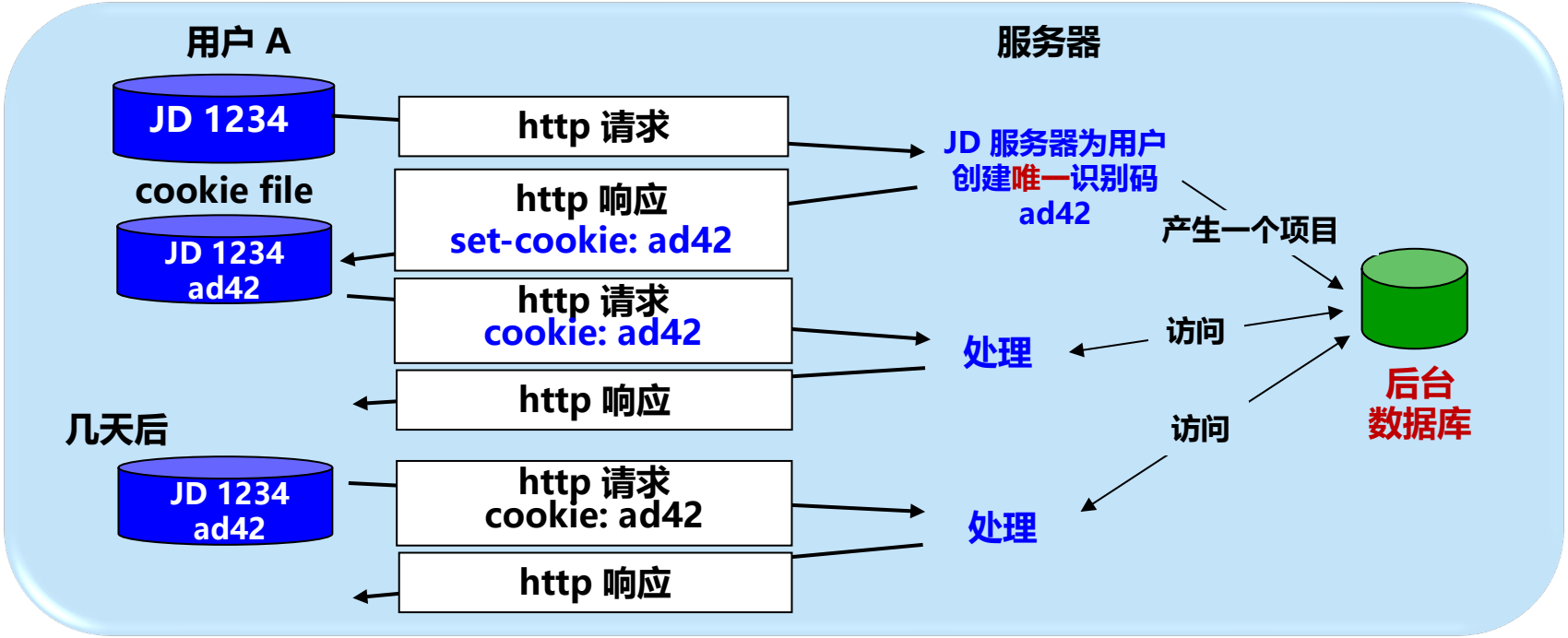
- 1xx 表示通知信息，如请求收到了或正在进行处理。
- 2xx 表示成功，如接受或知道了。
- 3xx 表示重定向，表示要完成请求还必须采取进一步的行动。
- 4xx 表示客户的差错，如请求中有错误的语法或不能完成。
- 5xx 表示服务器的差错，如服务器失效无法完成请求。

响应报文中常见到的三种状态行：

HTTP/1.1 202 Accepted	接受
HTTP/1.1 400 Bad Request	错误的请求
Http/1.1 404 Not Found	找不到

4. 在服务器上存放用户的信息

- 万维网使用 Cookie 跟踪在 HTTP 服务器和客户之间传递的状态信息。



6.4.4 万维网的文档

- 在一个客户程序主窗口上显示出的万维网文档称为**页面** (page)。
- 页面制作的标准语言：**HTML**。
- 分为：
  - **静态**万维网文档。内容不会改变。简单。
  - **动态**万维网文档。文档的内容由应用程序动态创建。
  - **活动**万维网文档。由浏览器端改变文档的内容。

1. 超文本标记语言 HTML

- 超文本标记语言 HTML (HyperText Markup Language) 是一种制作万维网页面的**标准语言**，它消除了不同计算机之间信息交流的障碍，是万维网的重要基础 [RFC 2854]。
- 最新 HTML 5.0 增加了<audio>和<video>两个标签，实现对多媒体中的音频、视频使用的支持，增加了能够在移动设备上支持多媒体功能。

注意：HTML **不是**应用层的协议，它只是万维网**浏览器使用的一种语言**。

1. 超文本标记语言 HTML

- HTML 定义了许多用于**排版的命令**（即**标签**）。
- HTML 把各种标签嵌入到万维网的页面中，构成了所谓的 HTML 文档。
- HTML 文档是一种可以用任何文本编辑器创建的 **ASCII 码文件。**
- HTML 文档的**后缀**：.html 或 .htm。

HTML 文档中标签的用法

文档

文档首部

```
<HTML>
<HEAD>
    <TITLE>一个 HTML 的例子</TITLE>
</HEAD>
```

文档主体

```
<BODY>
    <H1>HTML 很容易掌握</H1>
    <P>这是第一个段落。 </P>
    <P>这是第二个段落。 </P>
</BODY>
</HTML>
```

HTML 文档中标签的用法

文档

<HTML>

文档首部

<HEAD>
<TITLE>一个 HTML 的例子</TITLE>
</HEAD>

文档主体

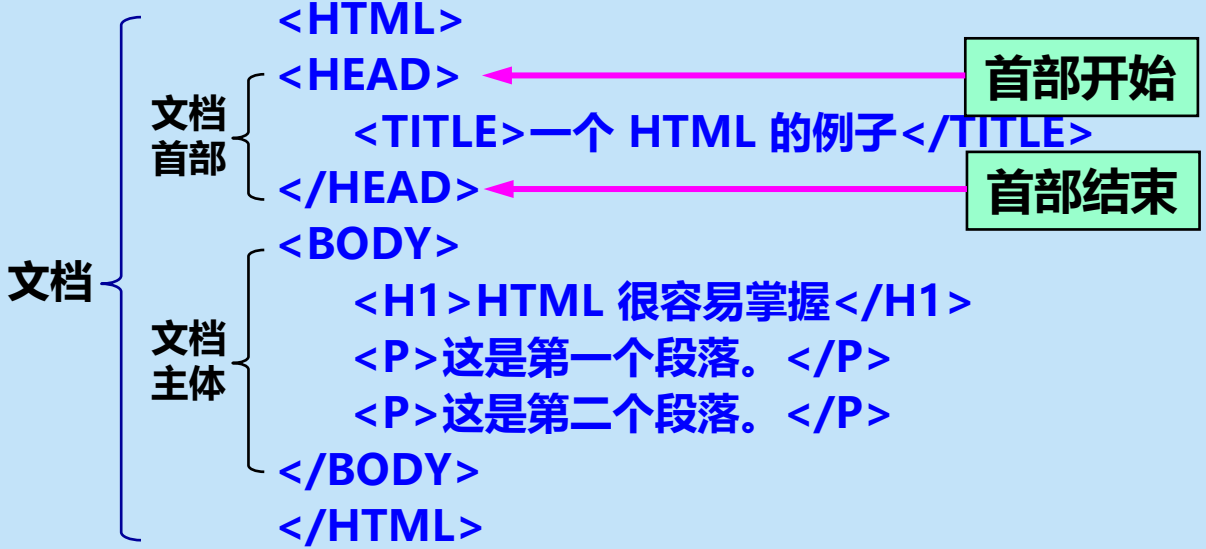
<BODY>
<H1>HTML 很容易掌握</H1>
<P>这是第一个段落。 </P>
<P>这是第二个段落。 </P>
</BODY>

</HTML>

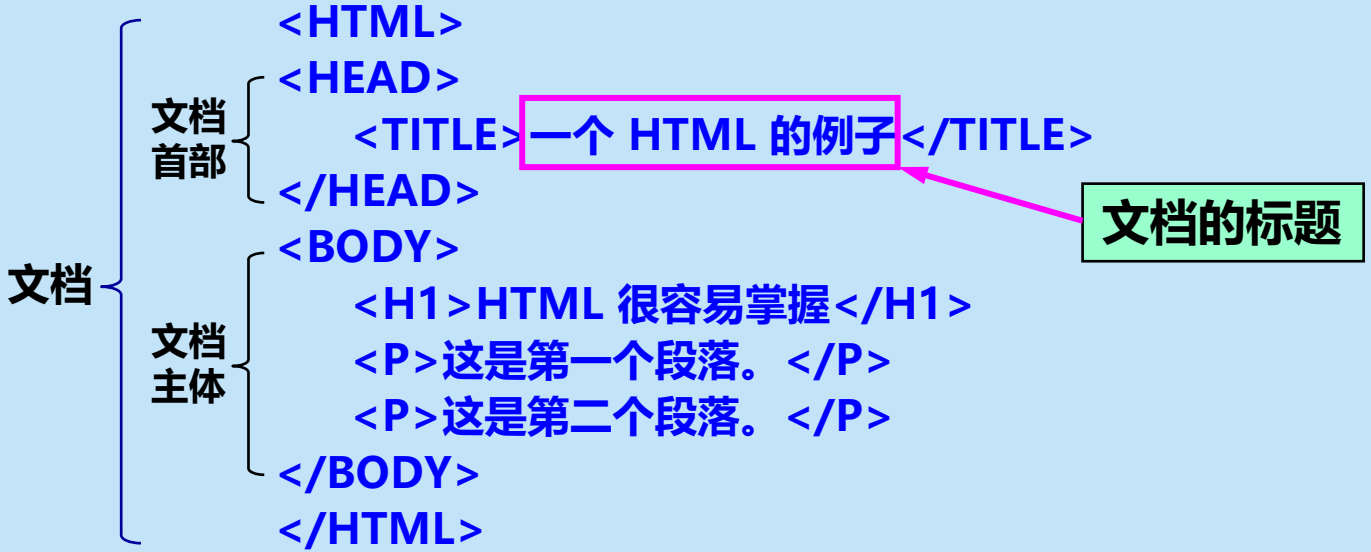
HTML 文档开始

HTML 文档结束

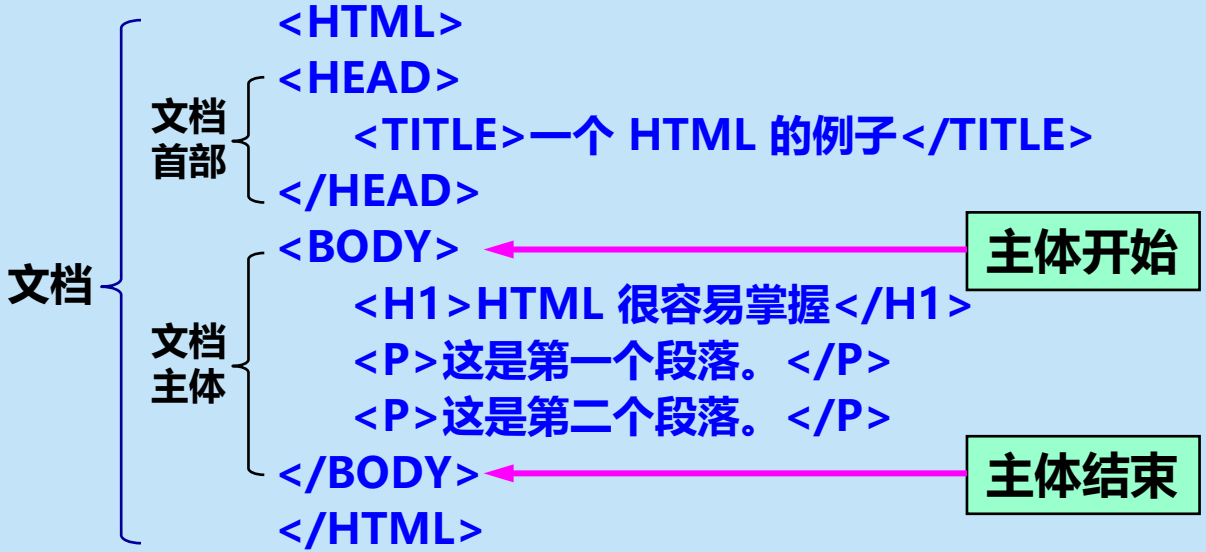
HTML 文档中标签的用法



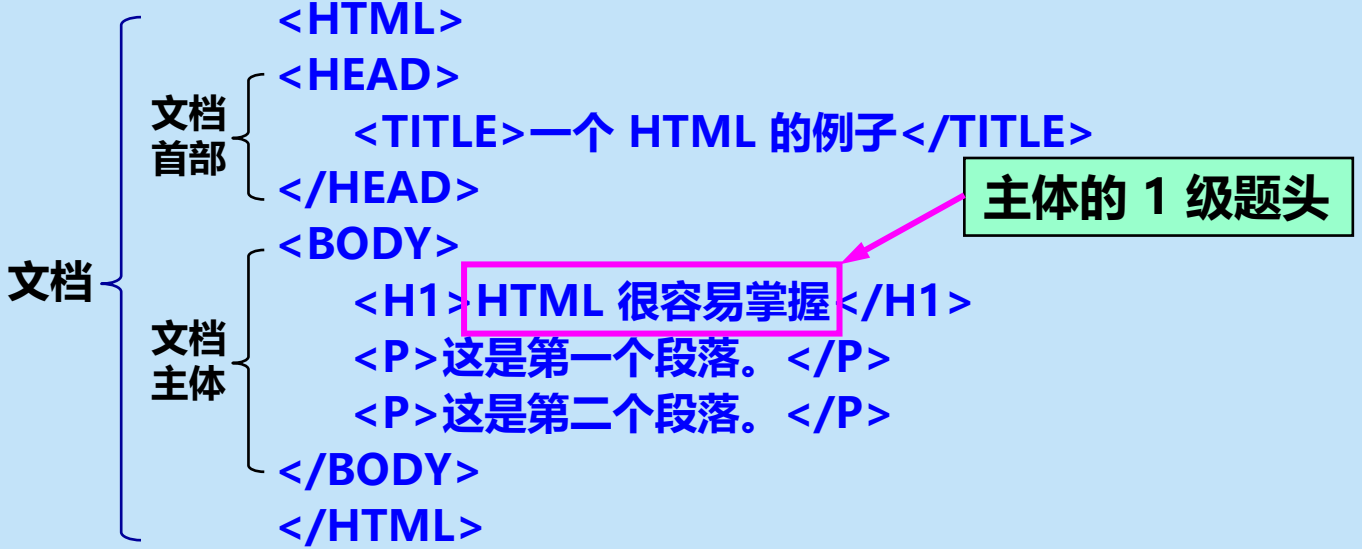
HTML 文档中标签的用法



HTML 文档中标签的用法



HTML 文档中标签的用法



HTML 文档中标签的用法

文档
- 文档首部
  - <HTML>
  - <HEAD>
  - <TITLE>一个 HTML 的例子</TITLE>
  - </HEAD>
- 文档主体
  - <BODY>
  - <H1>HTML 很容易掌握</H1>
  - <P>这是第一个段落。</P>
  - <P>这是第二个段落。</P>
  - </BODY>
  - </HTML>

主体的第一个段落

HTML 文档中标签的用法

文档
- 文档首部
  - <HTML>
  - <HEAD>
  - <TITLE>一个 HTML 的例子</TITLE>
  - </HEAD>
- 文档主体
  - <BODY>
  - <H1>HTML 很容易掌握</H1>
  - <P>这是第一个段落。 </P>
  - <P>这是第二个段落。</P>
  - </BODY>
  - </HTML>

主体的第二个段落

HTML 文档中标签的用法

- <img>：图像标签。
  - 允许在万维网页面中插入图像。
  - 一个页面本身带有的图像称为内含图像 (inline image)。
  - HTML 标准没有规定该图像的格式。
  - 大多数浏览器都支持 GIF 和 JPEG 文件。

HTML 文档中标签的用法

- <a>：链接标签。
  - 每个链接都有一个起点和终点。
  - 起点：说明在万维网页面中的什么地方可引出一个链接。可以是一个字或几个字，或是一幅图，或是一段文字。例如：

    清华大学主页
  - 终点：
    - 可以是其他网站上的页面。这种链接方式叫做远程链接。这时必须指明链接到的网站的 URL。
    - 可以指向本计算机中的某一个文件或本文件中的某处，这种链接方式叫做本地链接。这时必须指明链接的路径。

XML

- 可扩展标记语言 XML (Extensible Markup Language) 和 HTML 很相似。
- 设计宗旨是：传输数据，而不是显示数据。
- 特点和优点：
  - 可用来标记数据、定义数据类型；
  - 允许用户对自己的标记语言进行自定义，并且是无限制的；
  - 简单，与平台无关；
  - 将用户界面与结构化数据分隔开来；

XHTML

- 可扩展超文本标记语言 XHTML (Extensible HTML) 与 HTML 4.01 几乎相同，是更严格的 HTML 版本。
- 作为一种 XML 应用被重新定义的 HTML，将逐渐取代 HTML。

CSS

- **层叠样式表 CSS** (Cascading Style Sheets) 是一种样式表语言，用于为 HTML 文档**定义布局。**
- CSS 与 HTML 的**区别：**HTML 用于结构化内容，而 CSS 则用于格式化结构化的内容。
- 例如：精确规定在浏览器上显示的字体、颜色、边距、高度、宽度、背景图像等。

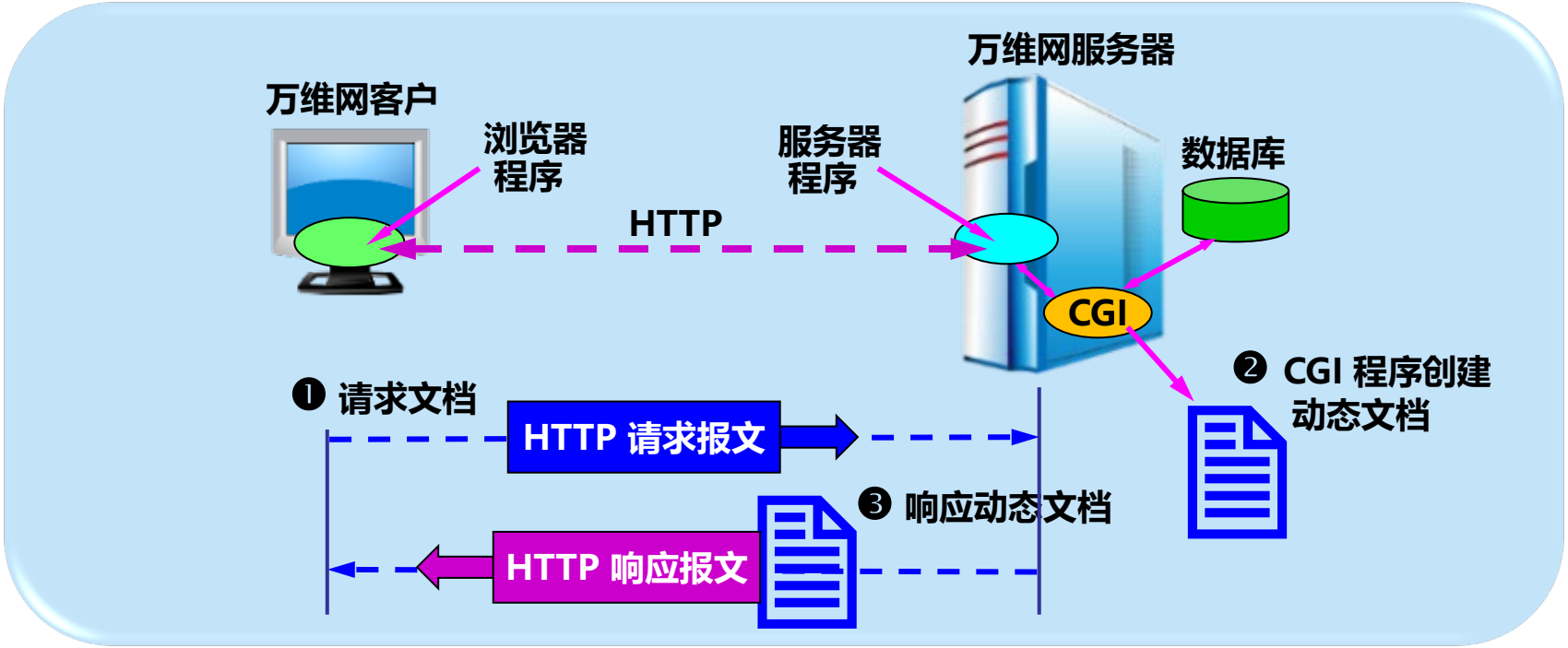
2. 动态万维网文档

- **静态文档：**该文档创作完毕后就存放在万维网服务器中，在被用户浏览的过程中，内容不会改变。
- **动态文档：**文档的内容是在浏览器访问万维网服务器时才由应用程序动态创建。
- 动态文档和静态文档之间的**主要差别**体现在**服务器端：**文档内容的生成方法不同。从浏览器的角度看，这两种文档并没有区别。

万维网服务器功能的扩充

- (1) 增加一个**应用程序**：处理浏览器发来的数据，并创建动态文档。
- (2) 增加一个**机制**：使万维网服务器把浏览器发来的数据**传送**给这个应用程序，然后万维网服务器能够**解释**这个应用程序的输出，并向浏览器**返回** HTML 文档。

扩充了功能的万维网服务器



CGI

- **通用网关接口 CGI** (Common Gateway Interface)：定义动态文档应如何创建，输入数据应如何提供给应用程序，以及输出结果应如何使用的一种标准。
- **通用**：CGI 标准所定义的规则对其他任何语言都是通用的。
- **网关**：CGI 程序的作用像网关。
- **接口**：有一些已定义好的变量和调用等可供其他 CGI 程序使用。

CGI 网关程序

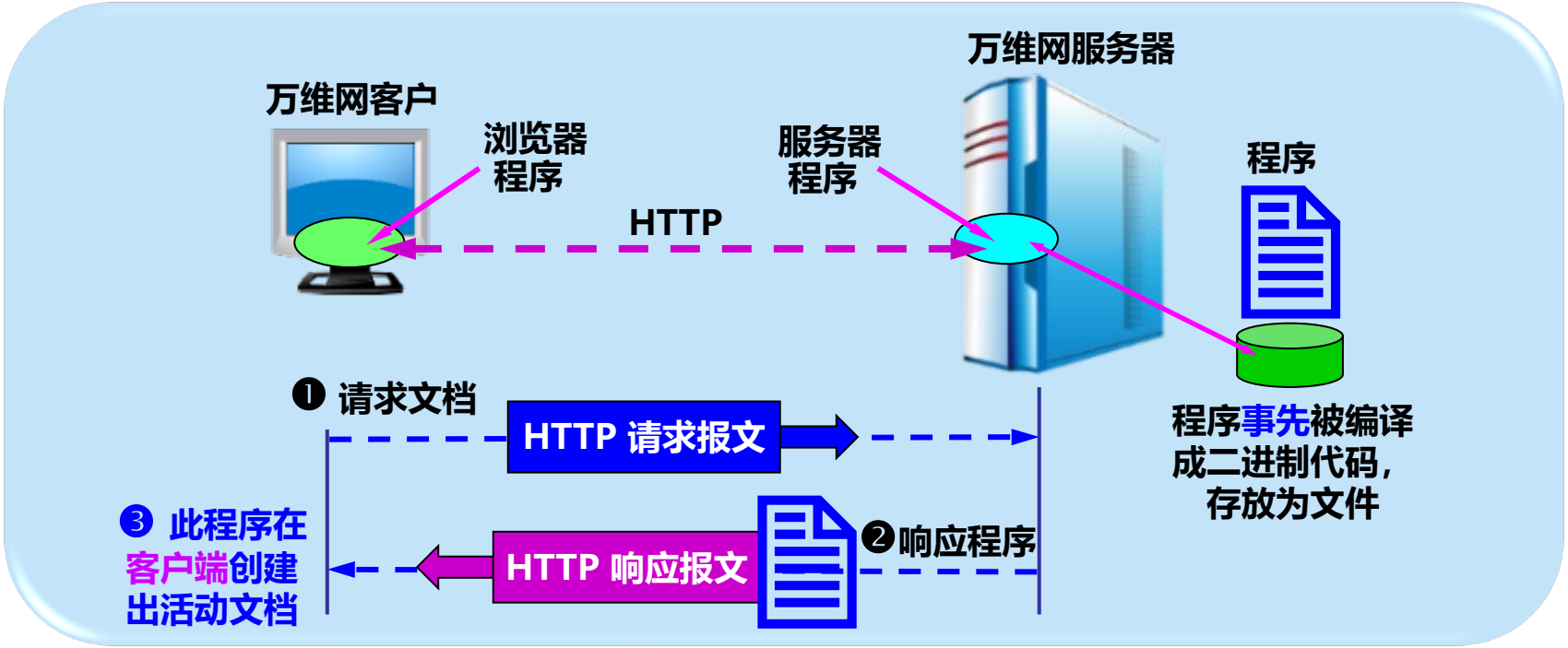
- 正式名字：**CGI 脚本** (script)。
- **脚本**：指的是一个程序，它被另一个程序（解释程序）而不是计算机的处理机来解释或执行。
- **脚本语言** (script language)：如 Perl, JavaScript，Tcl/Tk 等。也可用一些常用的编程语言写出，如 C，C++等。
- 脚本运行起来要比一般的编译程序要慢。
- 脚本不一定是一个独立的程序，可以是一个动态装入的库，甚至是服务器的一个子程序。

CGI 程序又称为 cgi-bin 脚本，因为在许多万维网服务器上，将 CGI 程序放在 /cgi-bin 的目录下。

3. 活动万维网文档

- **活动文档 (active document) 技术：把屏幕连续更新的工作转移给浏览器端。**
- **每当浏览器请求一个活动文档时，服务器就返回一段程序副本在浏览器端运行。**
- **活动文档程序可与用户直接交互，并可连续地改变屏幕的显示。**
- **由于活动文档技术不需要服务器的连续更新传送，对网络带宽的要求也不会太高。**

活动文档在客户端创建


万维网客户
浏览器
程序
HTTP
服务器
程序
万维网服务器
程序
程序事先被编译
成二进制代码，
存放为文件
❶ 请求文档
HTTP 请求报文
❷ 响应程序
HTTP 响应报文
❸ 此程序在
客户端创建
出活动文档

用 Java 语言创建活动文档

- Java 语言是一项用于创建和运行活动文档的技术。
- 在 Java 技术中使用小应用程序 (applet) 来描述活动文档程序。
- 用户从万维网服务器下载嵌入了 applet 的 HTML 文档后，可在浏览器的屏幕上点击某个图像，就可看到动画效果，或在下拉式菜单中点击某个项目，就可看到计算结果。
- Java 技术是活动文档技术的一部分。

6.4.5 万维网的信息检索系统

- 使用户能够很方便地找到所需的信息。
- 在万维网中用来进行搜索的程序叫做**搜索引擎** (search engine)。

1．全文检索搜索和分类目录搜索

全文检索搜索引擎

- 一种纯技术型的检索工具。
- 通过搜索软件（例如一种叫做“蜘蛛”或“网络机器人”的 Spider 程序）到互联网上的各网站收集信息。
- 按照一定的规则建立一个很大的在线索引数据库。
- 用户在查询时只要输入关键词，从已经建立的索引数据库里查询（非实时）。
- 需要定期更新维护数据库。

分类目录搜索引擎

- 不采集网站的任何信息，而是利用各网站向搜索引擎提交的网站信息时填写的关键词和网站描述等信息，经过人工审核编辑后，输入到分类目录的数据库中，供网上用户查询。
- 分类目录搜索也叫做分类网站搜索。
- 查询时只需要按照分类，不需要使用关键词，查询的准确性较好。
- 查询的结果不是具体的页面，而是被收录网站主页的 URL 地址。

(a) 全文检索举例

(b) 分类检索举例

一些著名的搜索引擎

- 全文检索搜索引擎：
  1. Google（谷歌）(www.google.com)
  2. 必应 (cn.bing.com)
  3. 百度 (www.baidu.com)。
- 分类目录搜索引擎：
  1. 雅虎 (www.yahoo.com)
  2. 雅虎中国 (cn.yahoo.com)
  3. 新浪 (www.sina.com)
  4. 搜狐 (www.sohu.com)
  5. 网易 (www.163.com)

垂直搜索引擎

- **垂直搜索引擎** (Vertical Search Engine)：
  - 针对某一特定领域、特定人群或某一特定需求提供搜索服务。
  - 也是提供关键字来进行搜索，但被放到一个行业知识的上下文中，返回的结果更倾向于信息、消息、条目等。
  - 目前热门的垂直搜索行业有：购物、旅游、汽车、求职、房产、交友等。

元搜索引擎

- **元搜索引擎 (Meta Search Engine)：**
  - 把用户提交的检索请求发送到多个独立的搜索引擎上去搜索，并把检索结果集中统一处理，以统一的格式提供给用户，因此是搜索引擎之上的搜索引擎。
  - 主要精力放在提高搜索速度、智能化处理搜索结果、个性化搜索功能的设置和用户检索界面的友好性上。
  - 其查全率和查准率都比较高。

2. Google 搜索技术的特点

- **核心技术：网页排名（PageRank）**
  - **对搜索结果按重要性排序。**
  - **对链接的数目进行加权统计。来自重要网站的链接，其权重较大。**
  - **进行超文本匹配分析，确定哪些网页与正在执行的特定搜索相关。**
  - **在综合考虑整体重要性以及与特定查询的相关性之后，Google 就把最相关、最可靠的搜索结果放在首位。**

6.4.6 博客和微博

1. 博客

- 万维网日志 (weblog) 的简称。
- 使网民不仅是互联网上内容的消费者，而且还是互联网上内容的生产者。

2. 微博

- 微型博客 (microblog)，又称为微博客。
- 只记录片段、碎语，三言两语，现场记录，发发感慨，晒晒心情，永远只针对一个问题进行回答。
- 开通的多种 API 使用户可通过手机、网络等方式即时更新自己的信息。
- 是一种互动及传播性极快的工具。

6.4.7 社交网站

- **社交网站 SNS** (Social Networking Site)：为一群拥有相同兴趣与活动的人创建在线社区。
- 功能丰富：如电子邮件、即时传信（在线聊天）、博客撰写、共享相册、上传视频、网页游戏、创建社团、刊登广告等。
- 热门：
  - 脸书 (Facebook，又名面书、脸谱、脸谱网) (Facebook.com)。
  - YouTube.com
  - 推特 Twitter (twitter.com) 。
  - 微信 (weixin.qq.com)。
  - 抖音

6.5 电子邮件

6.5.1 电子邮件概述

6.5.2 简单邮件传送协议 SMTP

6.5.3 电子邮件的信息格式

6.5.4 邮件读取协议 POP3 和 IMAP

6.5.5 基于万维网的电子邮件

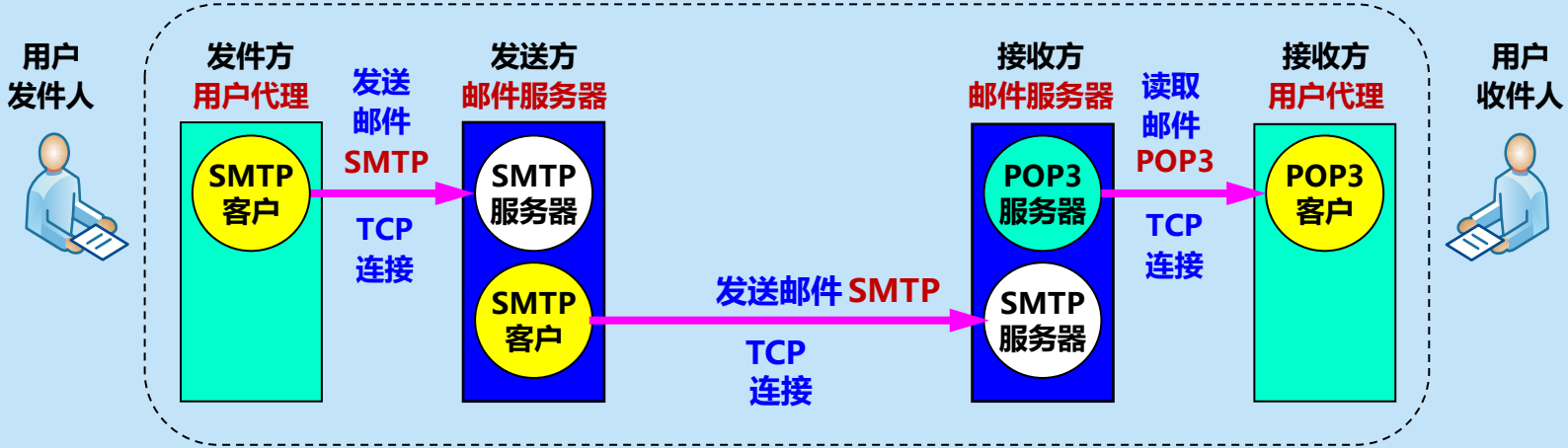
6.5.6 通用互联网邮件扩充 MIME

6.5.1 电子邮件概述

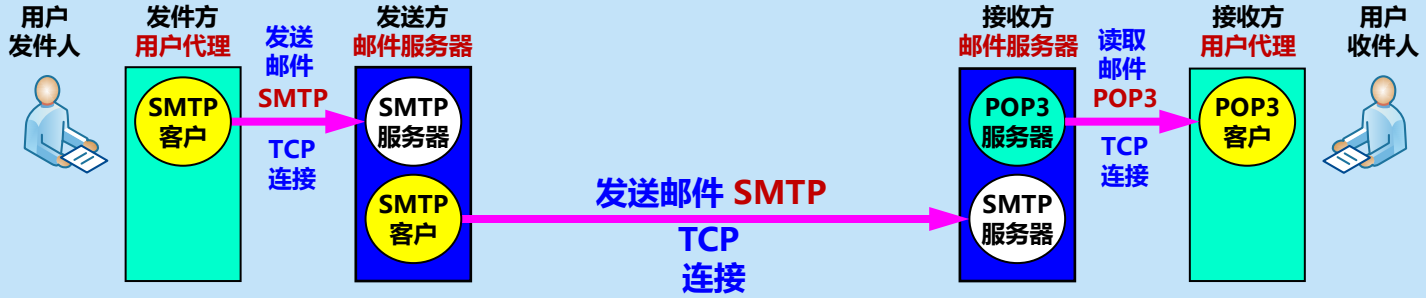
- **电子邮件** (e-mail)：指使用电子设备交换的邮件及其方法。
- **优点**：使用方便，传递迅速，费用低廉，可以传送多种类型的信息（包括：文字信息，声音和图像等）。
- **重要标准**：
  - 简单邮件发送协议：SMTP
  - 互联网文本报文格式
  - 通用互联网邮件扩充 MIME
  - 邮件读取协议：POP3 和 IMAP

电子邮件系统的组成：三个主要构件

用户代理，邮件服务器，以及邮件发送和读取协议。



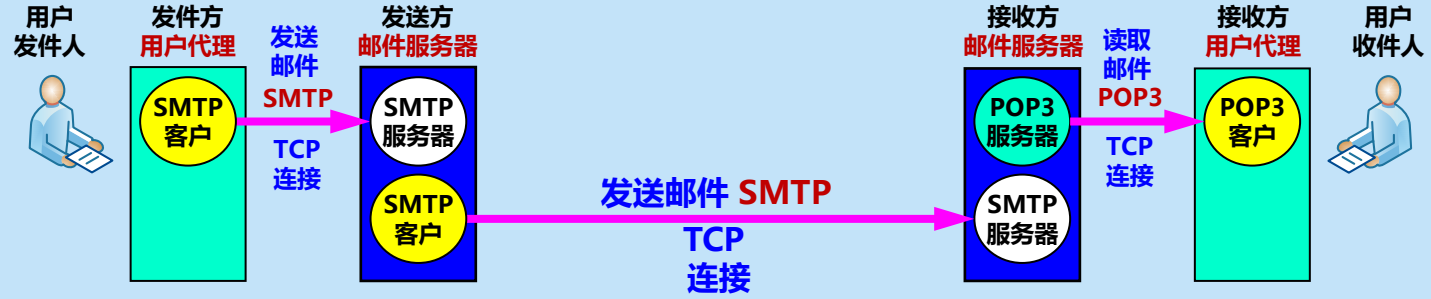
用户代理 UA (User Agent)



用户与电子邮件系统的接口。又被称为**电子邮件客户端软件。**

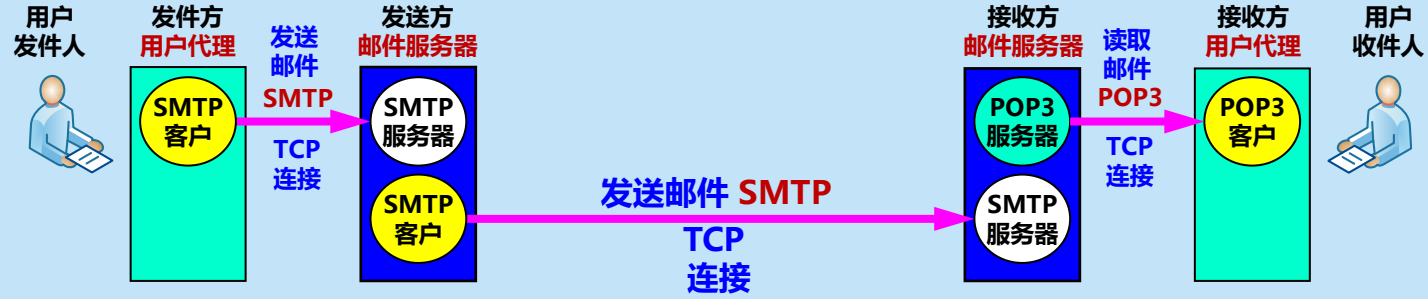
基本功能：撰写、显示、处理、通信。

邮件服务器 (Mail Server)



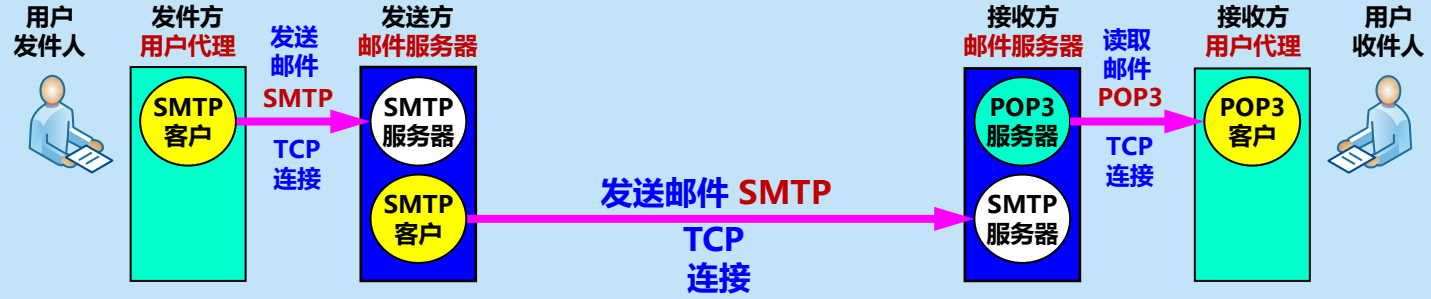
- 又被称为**邮件传输代理**。
- **功能**：发送和接收邮件，同时还要向发信人报告邮件传送的情况。
- 按照**客户服务器**方式工作。

邮件发送和读取协议



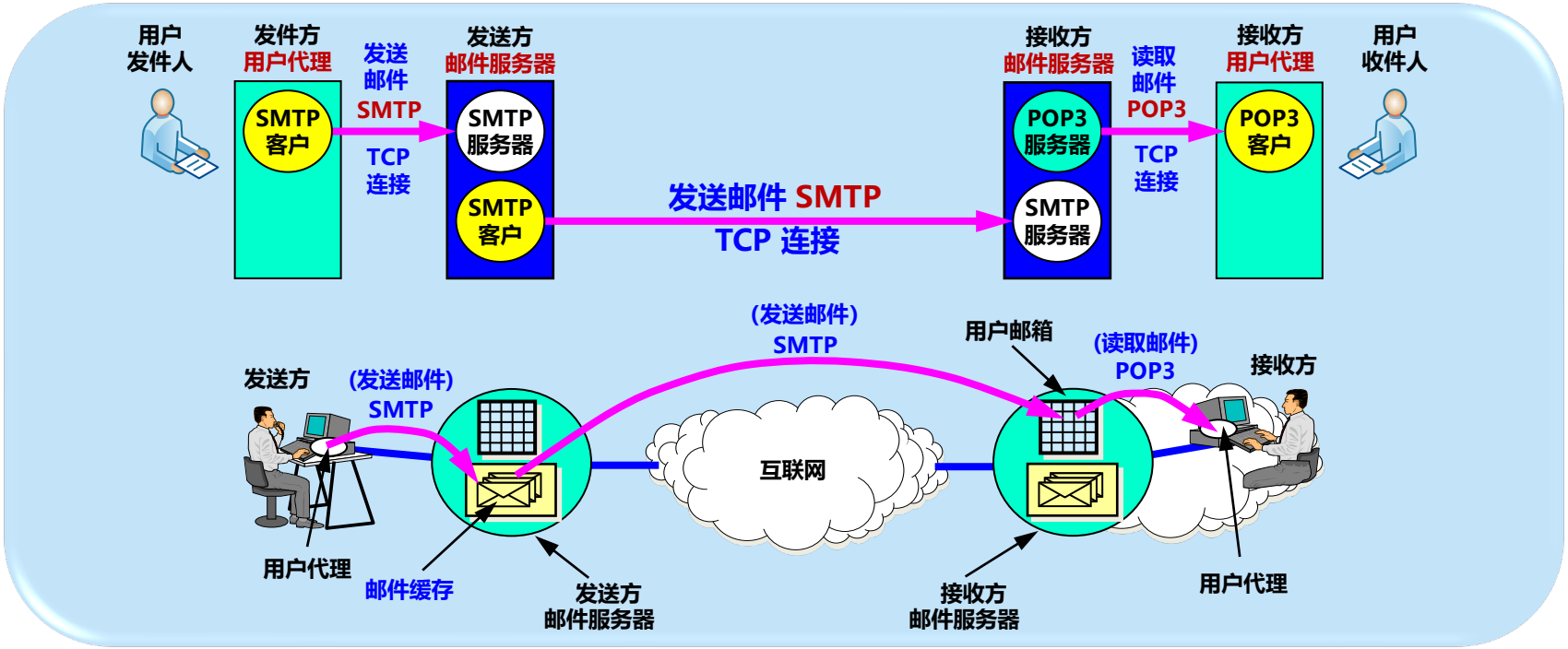
- 邮件发送和读取使用不同的协议。
- 简单邮件发送协议 SMTP：用于在用户代理向邮件服务器 或 邮件服务器之间发送邮件。
- 邮局协议 POP3：用于用户代理从邮件服务器读取邮件。

应当注意

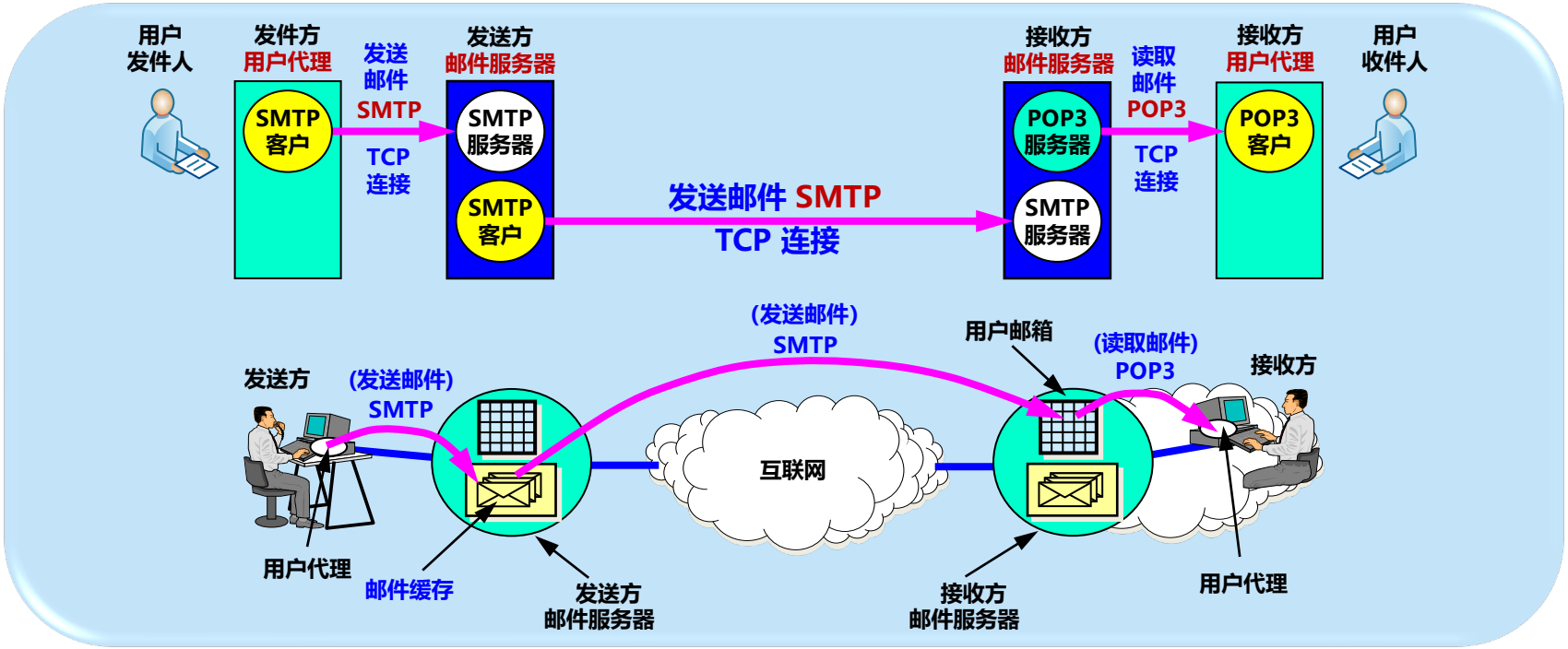


- 邮件服务器必须能够同时充当客户和服务器。
- SMTP 和 POP3（或 IMAP）都使用 TCP 连接可靠地传送邮件。

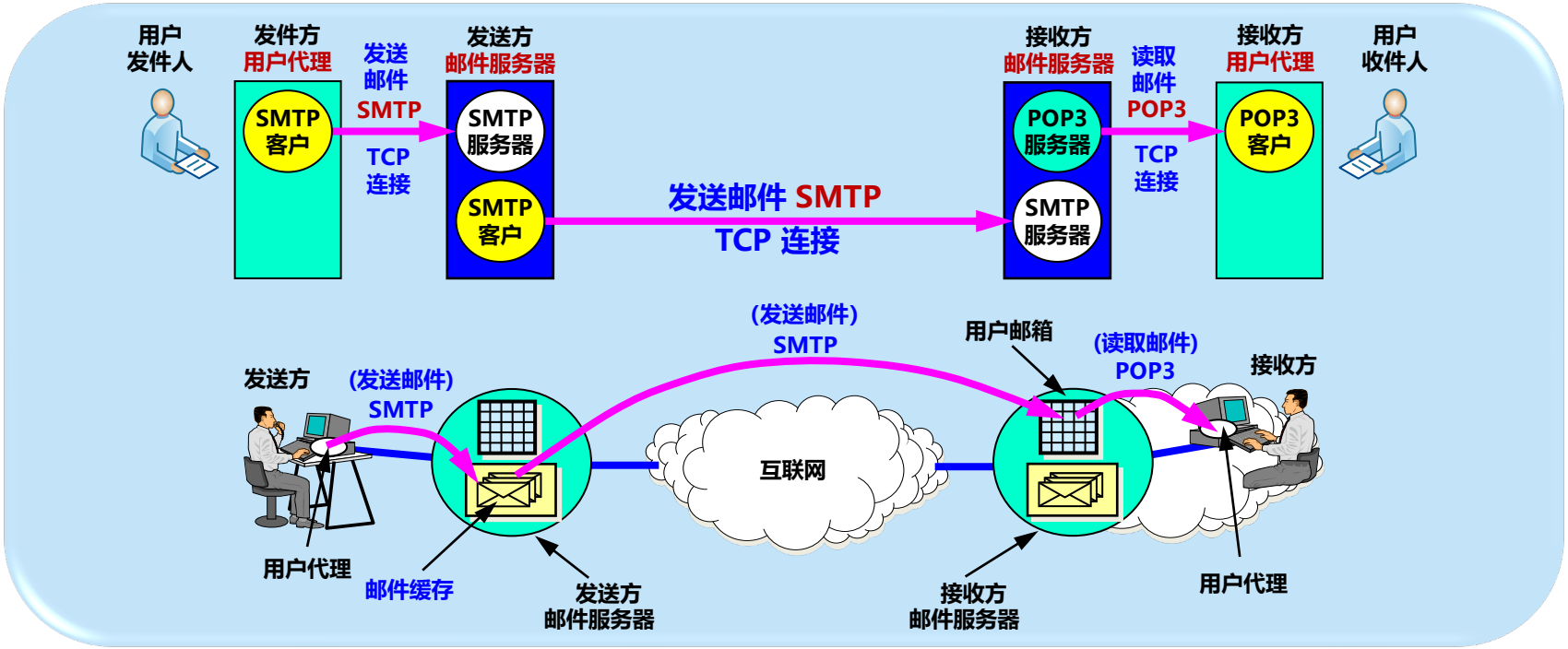
发送和接收电子邮件的重要步骤



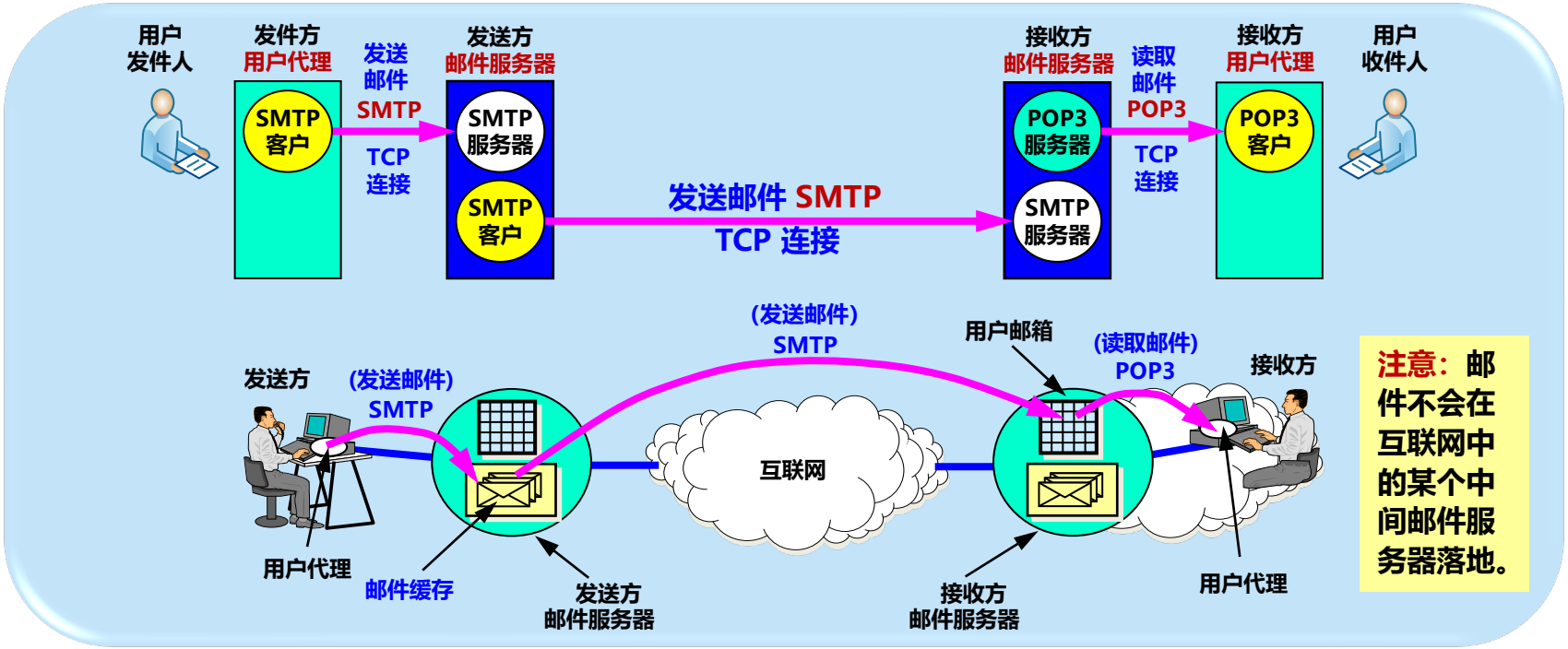
发送和接收电子邮件的重要步骤



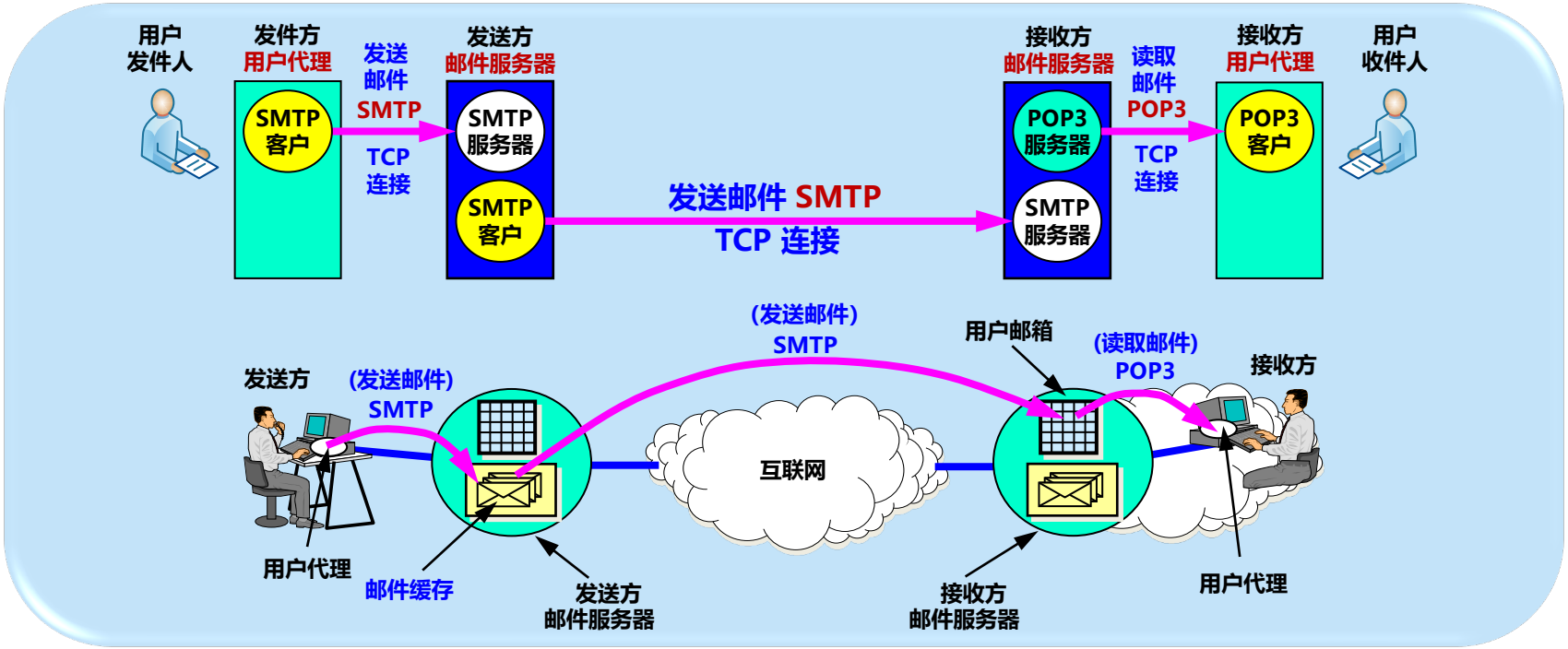
发送和接收电子邮件的重要步骤



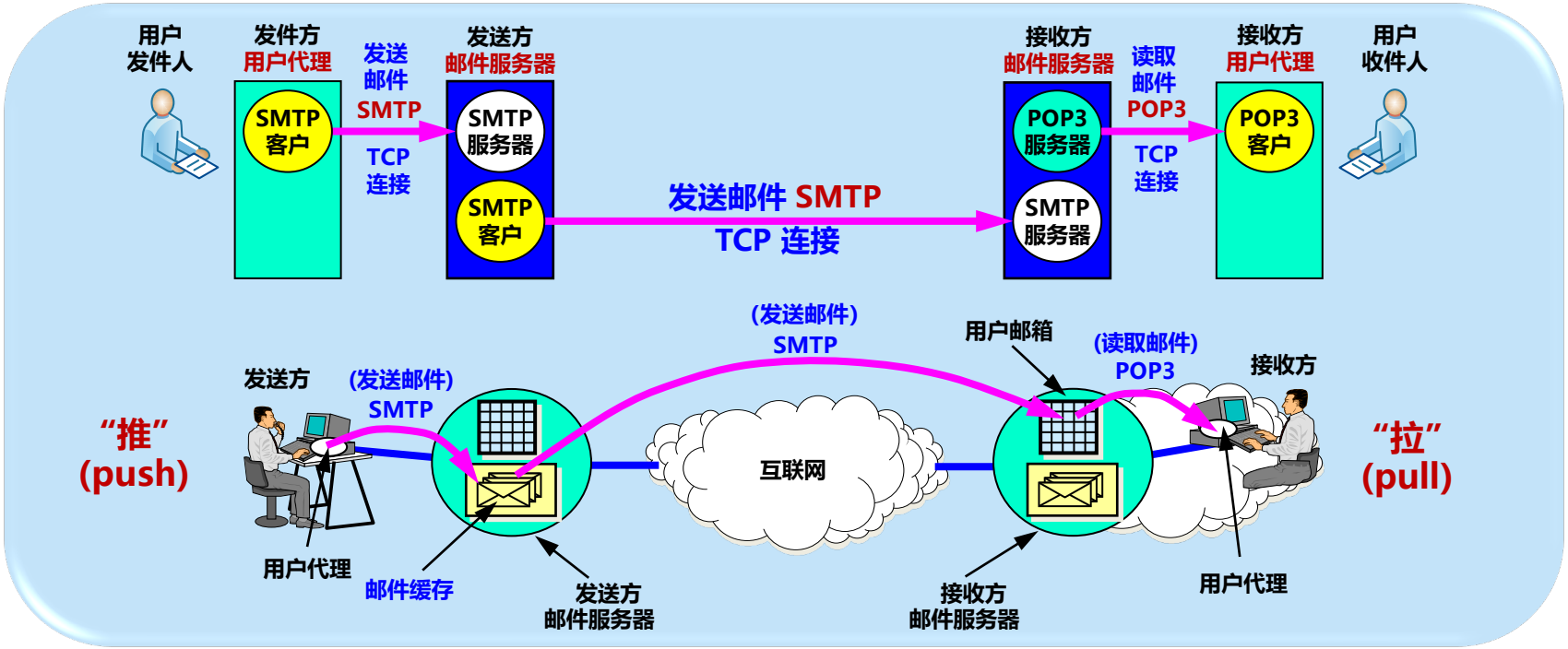
发送和接收电子邮件的重要步骤



发送和接收电子邮件的重要步骤



两种不同的通信方式


用户
发件人
发件方
用户代理
SMTP
客户
发送
邮件
SMTP
TCP
连接
发送方
邮件服务器
SMTP
服务器
SMTP
客户
发送邮件 SMTP
TCP 连接
接收方
邮件服务器
POP3
服务器
SMTP
服务器
读取
邮件
POP3
TCP
连接
接收方
用户代理
POP3
客户
用户
收件人
“推”
(push)
发送方
(发送邮件)
SMTP
用户代理
邮件缓存
发送方
邮件服务器
(发送邮件)
SMTP
互联网
用户邮箱
(读取邮件)
POP3
接收方
接收方
邮件服务器
用户代理
“拉”
(pull)

电子邮件的组成

- 电子邮件由信封 (envelope) 和内容 (content) 两部分组成。
- 电子邮件的传输程序根据邮件信封上的信息来传送邮件。
- 用户在从自己的邮箱中读取邮件时才能见到邮件的内容。

电子邮件地址的格式

- 在邮件的信封上，最重要的就是收件人的地址。
- TCP/IP 体系的电子邮件系统规定电子邮件地址的格式如下：

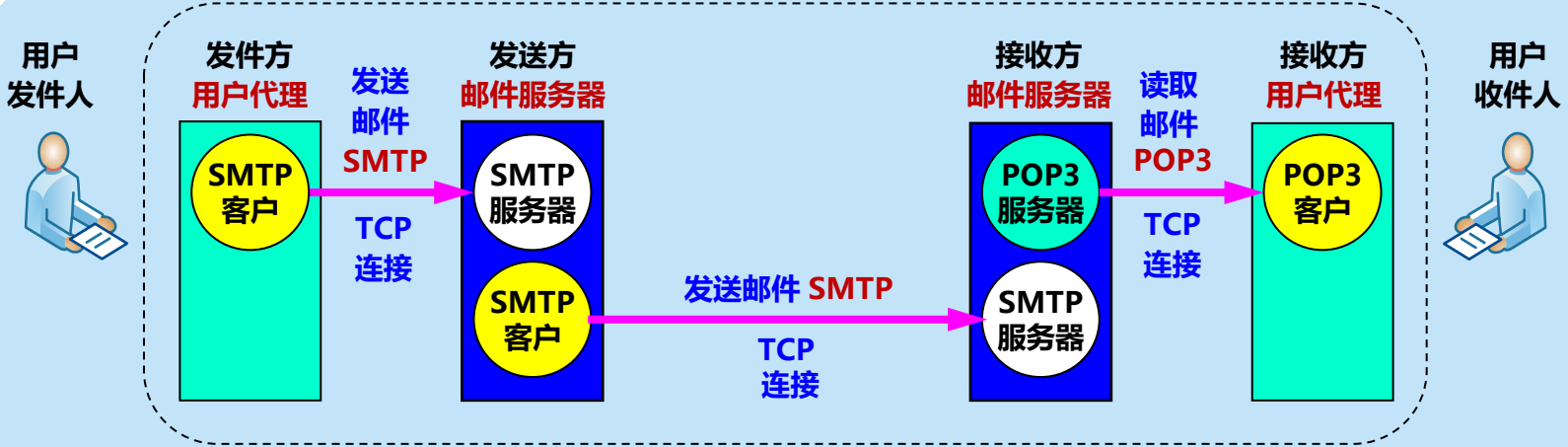
收件人邮箱名@邮箱所在主机的域名　　(6-1)

- 例如：　　xiexiren@tsinghua.org.cn

这个用户名在该域名的范围内是唯一的。

邮箱所在的主机的域名，在全世界必须是唯一的

6.5.2 简单邮件传送协议 SMTP



- SMTP 规定了在两个相互通信的 SMTP 进程之间交换信息的方法。
- SMTP 使用客户服务器方式。
- SMTP 基于 TCP 实现客户与服务器的通信。

6.5.2 简单邮件传送协议 SMTP

- SMTP 是一个基于文本的 （即 ASCII 码） 的协议。
- SMTP 客户与服务器之间采用命令-响应方式进行交互。

6.5.2 简单邮件传送协议 SMTP

SMTP 基于 TCP 实现客户与服务器之间的通信。

SMTP 通信的三个阶段

1. **连接建立：**连接是在发送主机的 SMTP 客户和接收主机的 SMTP 服务器之间建立的。SMTP 不使用中间的邮件服务器。
2. **邮件传送**
3. **连接释放：**邮件发送完毕后，SMTP 应释放 TCP 连接。

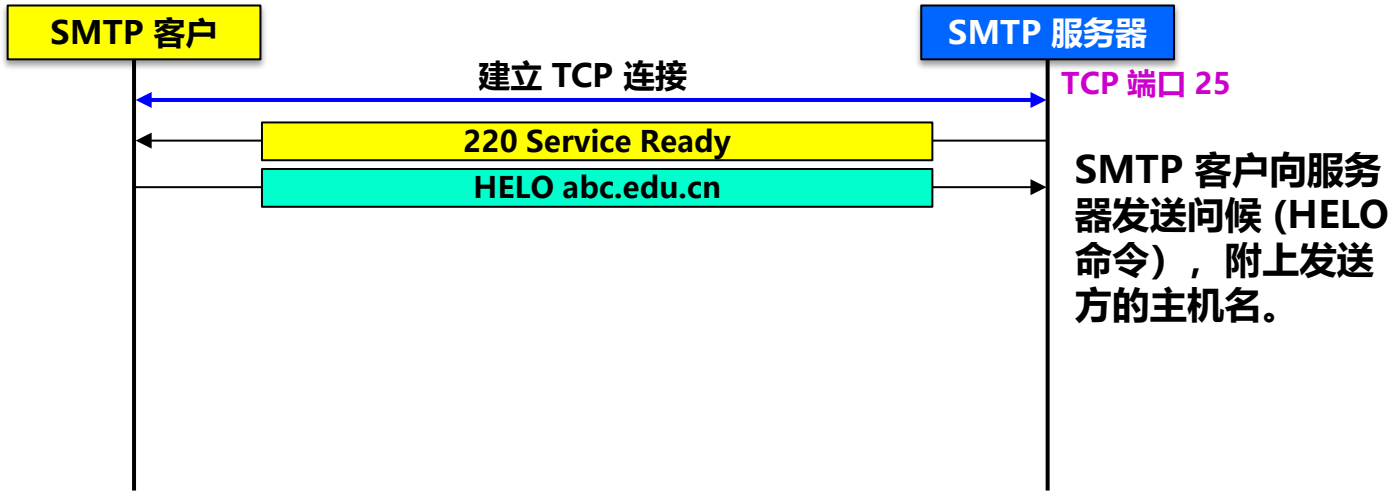
SMTP 通信的三个阶段

SMTP 通信的三个阶段

连接建立

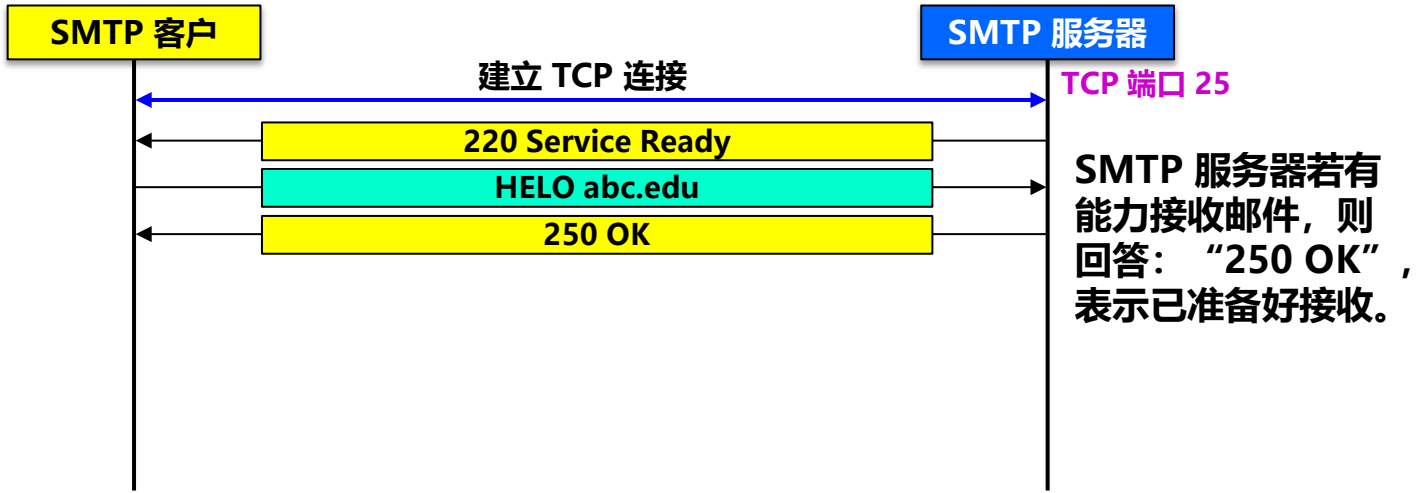
SMTP 通信的三个阶段

连接建立



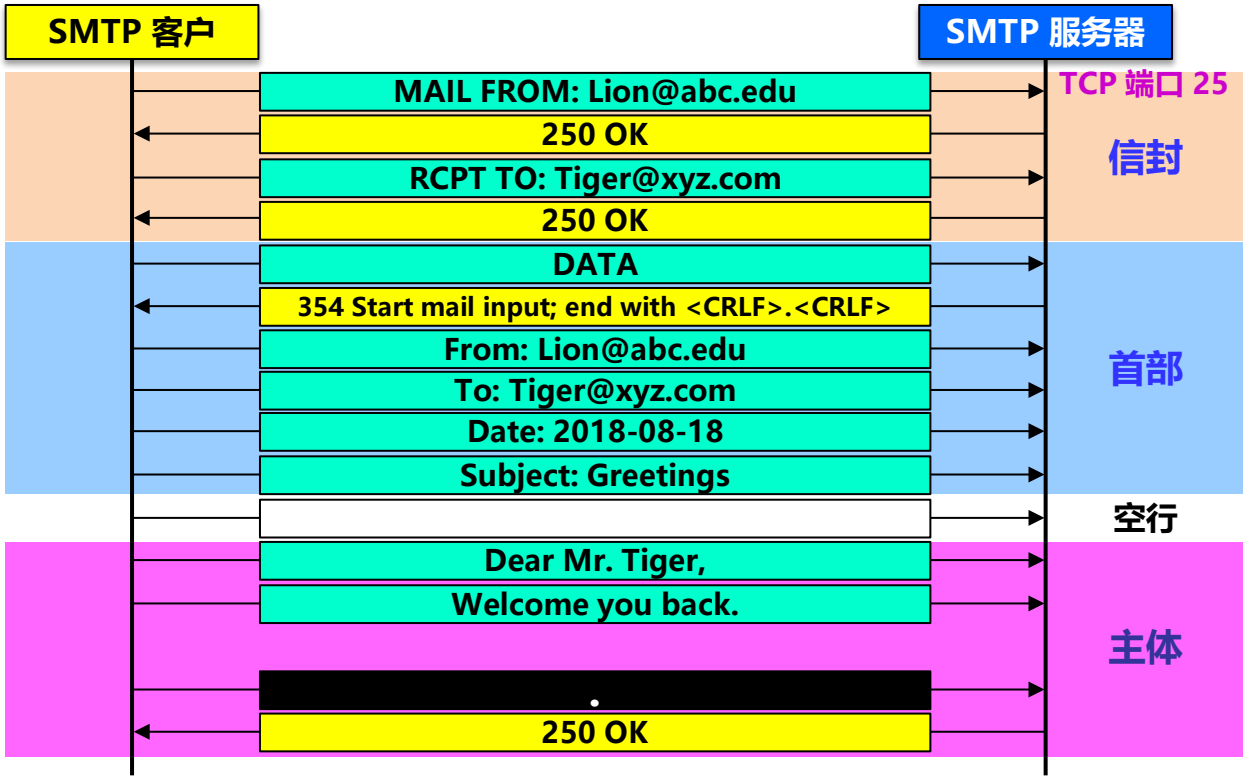
SMTP 通信的三个阶段

连接建立



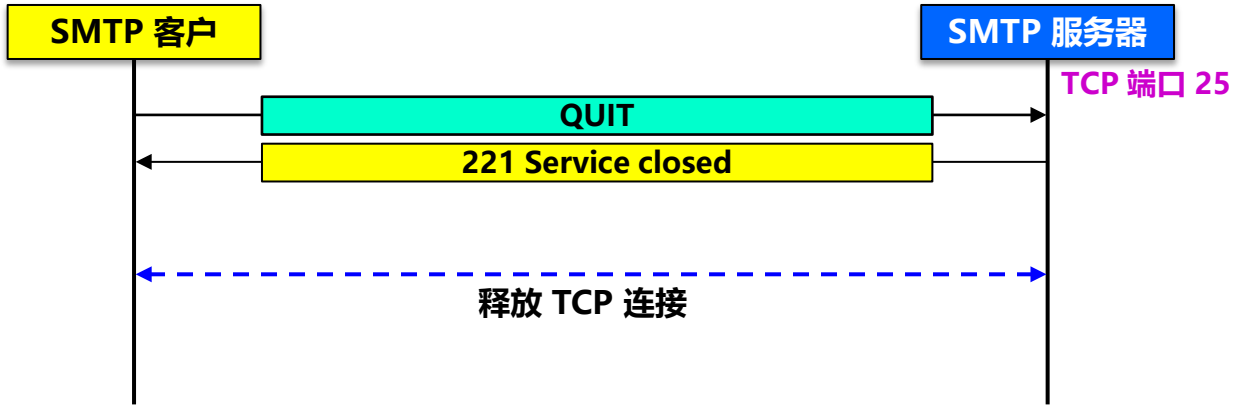
SMTP 通信的三个阶段

邮件传送



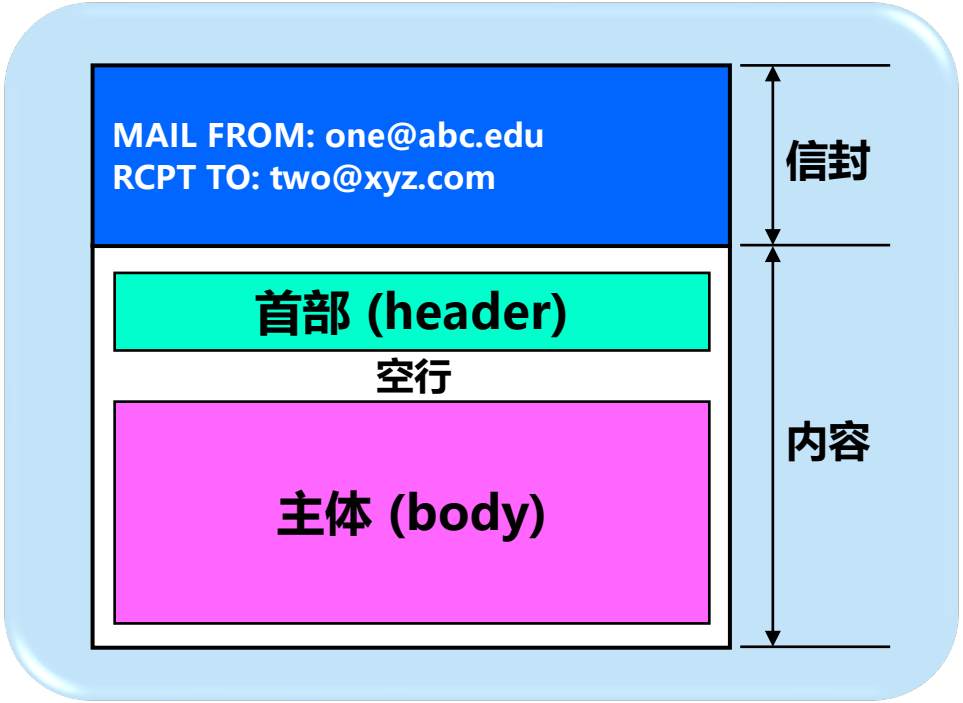
SMTP 通信的三个阶段

连接释放



6.5.3 电子邮件的信息格式

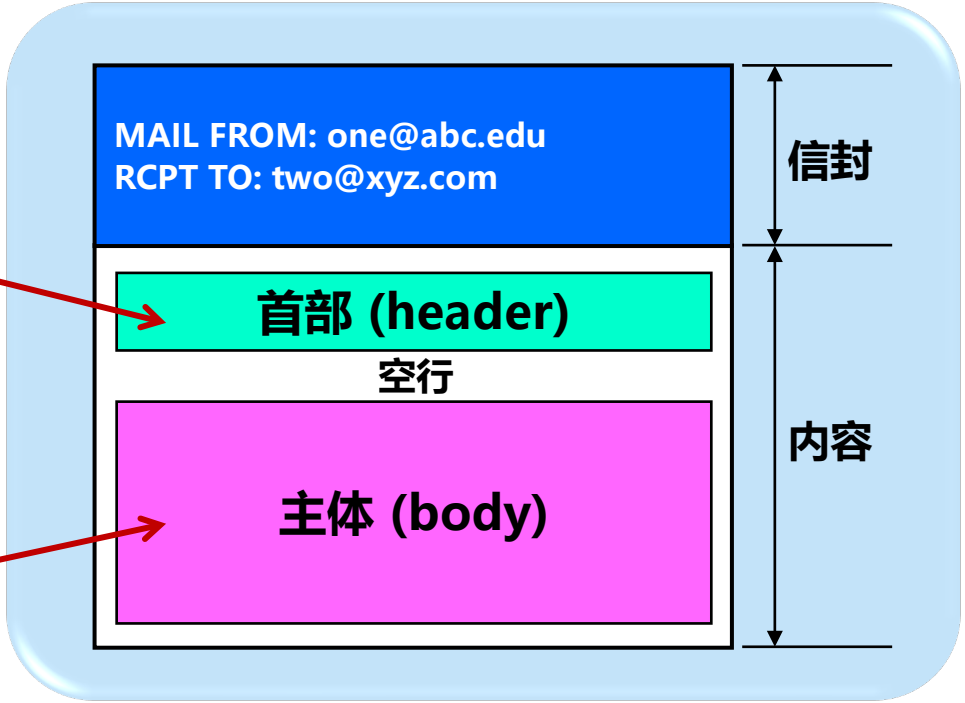
- 一个电子邮件分为信封和内容两大部分。
- RFC 5322 只规定了邮件内容中的首部 (header) 格式。
- 邮件的主体 (body) 部分则让用户自由撰写。



邮件内容的首部

- From: LionKing@abc.edu
- To: TigerWang@xyz.com
- Date: 2018-08-18
- Subject: Greetings
- Cc: Manager@zoo.com

- 纯文本信息 （ASCII编码）。
- 无特定结构和含义。

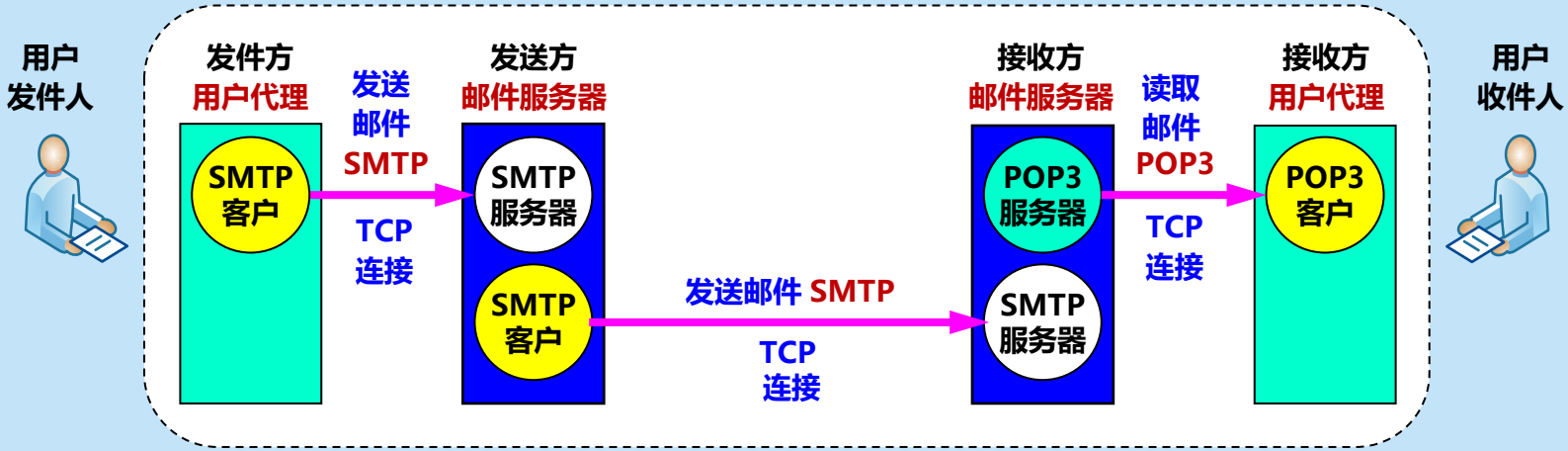


6.5.4 邮件读取协议 POP3 和 IMAP

两个常用的邮件读取协议：

1. POP3：邮局协议 (Post Office Protocol) 第3个版本
2. IMAP：网际报文存取协议 (Internet Message Access Protocol)

POP3 协议

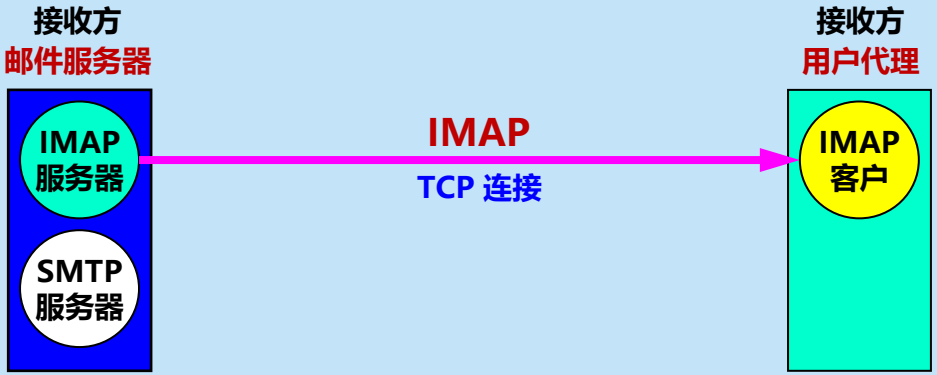


- POP3 使用客户服务器方式。
- POP3 基于 TCP 实现客户与服务器的通信。

POP3 协议

- POP3 支持用户鉴别。
- POP3 服务器删除被用户读取了的邮件。

IMAP 协议



- IMAP 使用客户服务器方式。
- IMAP 基于 TCP 实现客户与服务器的通信。
- IMAP 是一个联机协议。

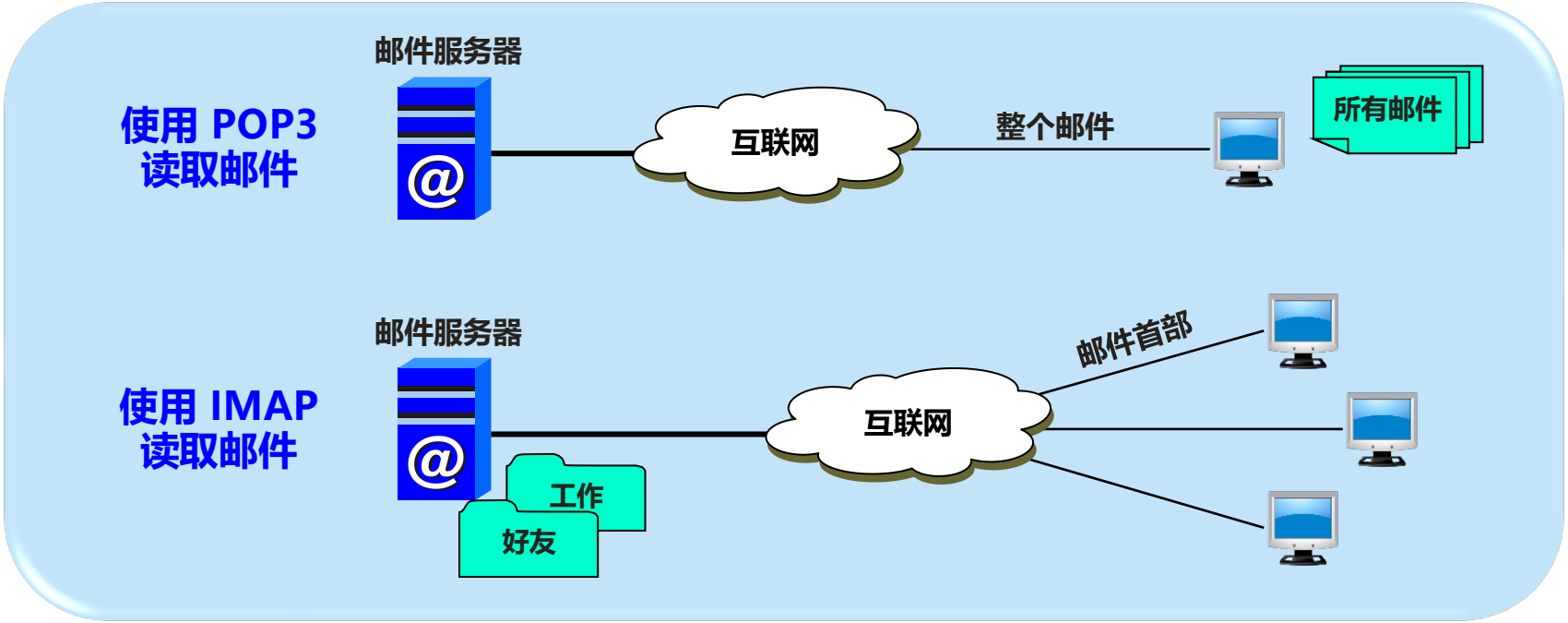
IMAP 的特点

- 连接后只下载**邮件首部**（部分下载）。
- 用户直接在 IMAP 服务器上**创建和管理**文件夹。
- 用户可以**搜索**邮件内容。
- 用户可以在不同的地方使用不同的计算机随时上网阅读和处理自己的邮件。
- 允许收信人**只读取**邮件中的某一个部分。
- **缺点**：要想查阅邮件，必须先联网。

IMAP 与 POP3 比较

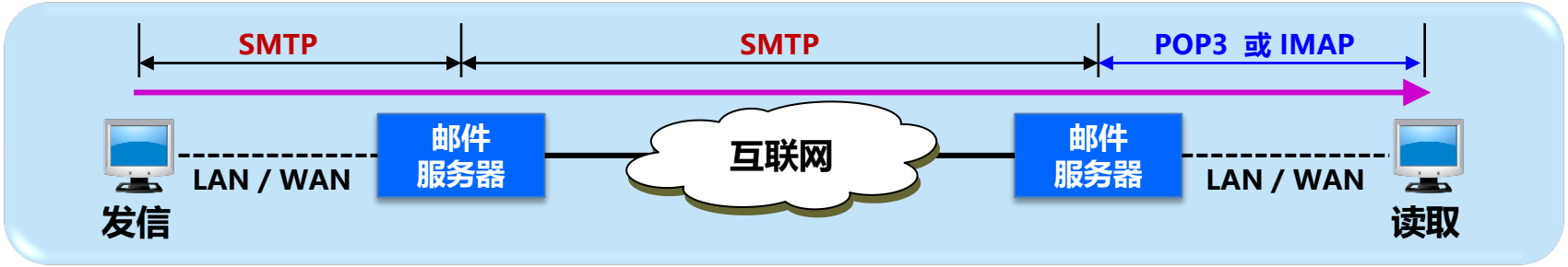
操作位置	操作内容	IMAP	POP3
收件箱	阅读、标记、移动、删除邮件等	客户端与邮箱更新同步	仅在客户端内
发件箱	保存到已发送	客户端与邮箱更新同步	仅在客户端内
创建文件夹	新建自定义的文件夹	客户端与邮箱更新同步	仅在客户端内
草稿	保存草稿	客户端与邮箱更新同步	仅在客户端内
垃圾文件夹	接收并移入垃圾文件夹的邮件	支持	不支持
广告邮件	接收并移入广告邮件夹的邮	支持	不支持

IMAP 与 POP3 比较



必须注意

- 邮件读取协议 POP 或 IMAP 与邮件传送协议 SMTP 完全不同。
- 发信人的用户代理向源邮件服务器发送邮件，以及源邮件服务器向目的邮件服务器发送邮件，都是使用 SMTP 协议。
- 而 POP 协议或 IMAP 协议则是用户从目的邮件服务器上读取邮件所使用的协议。



6.5.5 基于万维网的电子邮件

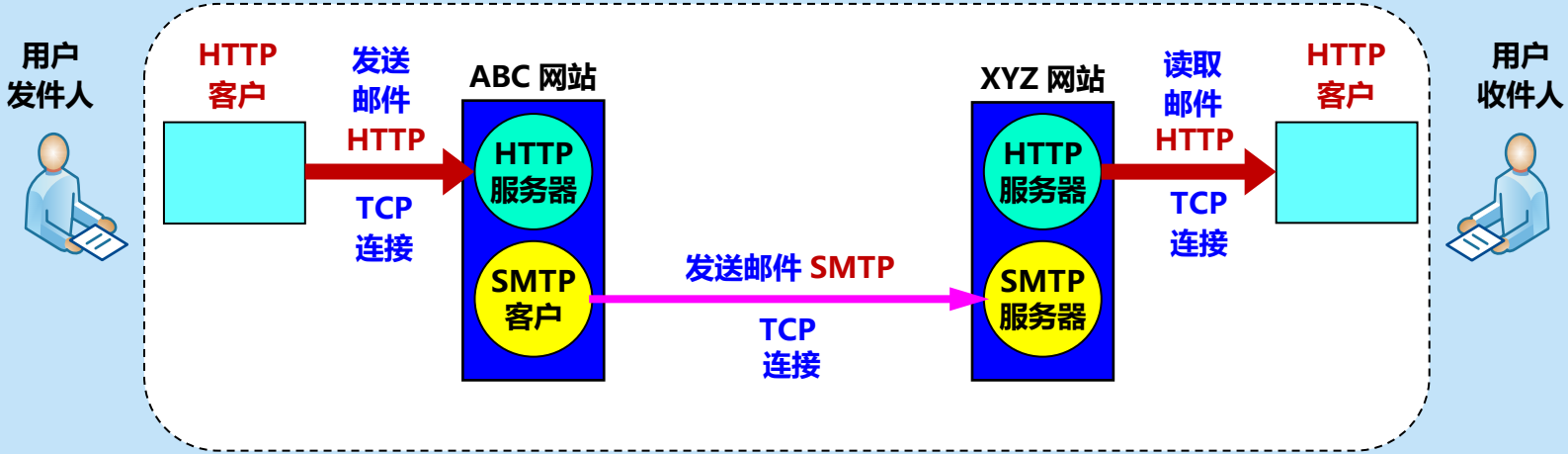
用户代理 (UA) 的缺点：

- 必须在计算机中安装用户代理软件。
- 收发邮件不方便。

万维网电子邮件优点：

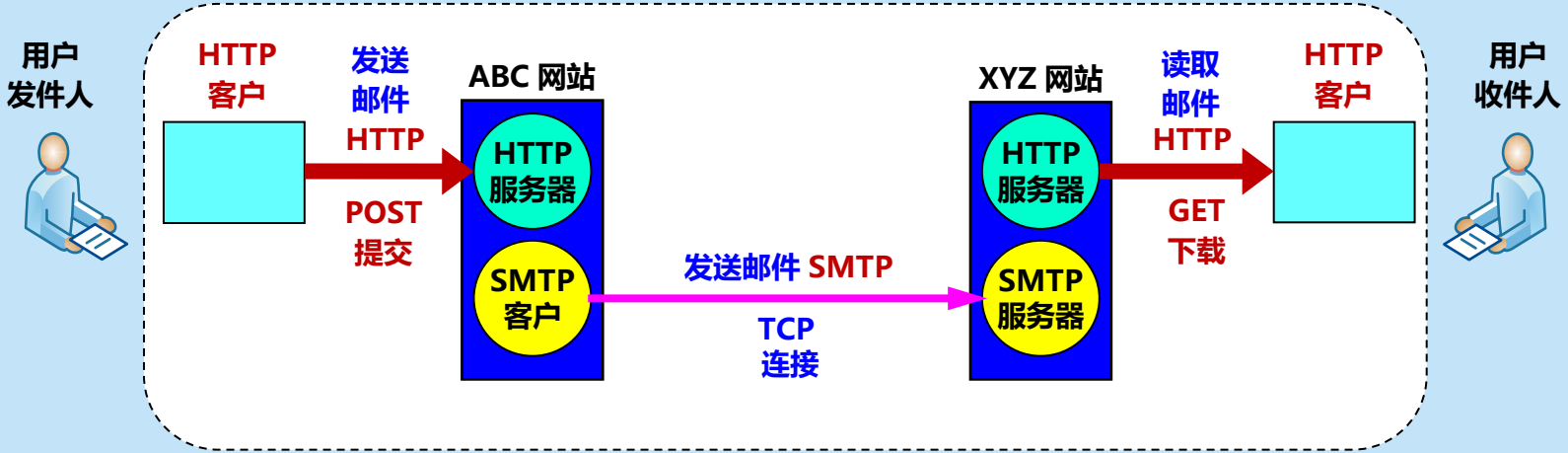
- 不需要在计算机中再安装用户代理软件。
- 计算机能联网，就能非常方便地收发电子邮件。
- 界面非常友好。

万维网电子邮件



- 发送、接收电子邮件时使用 HTTP 协议。
- 两个邮件服务器之间传送邮件时使用 SMTP。

万维网电子邮件



- 使用 HTTP POST 方法提交要发送的邮件。
- 使用 HTTP GET 方法读取邮件。

6.5.6 通用互联网邮件扩充 MIME

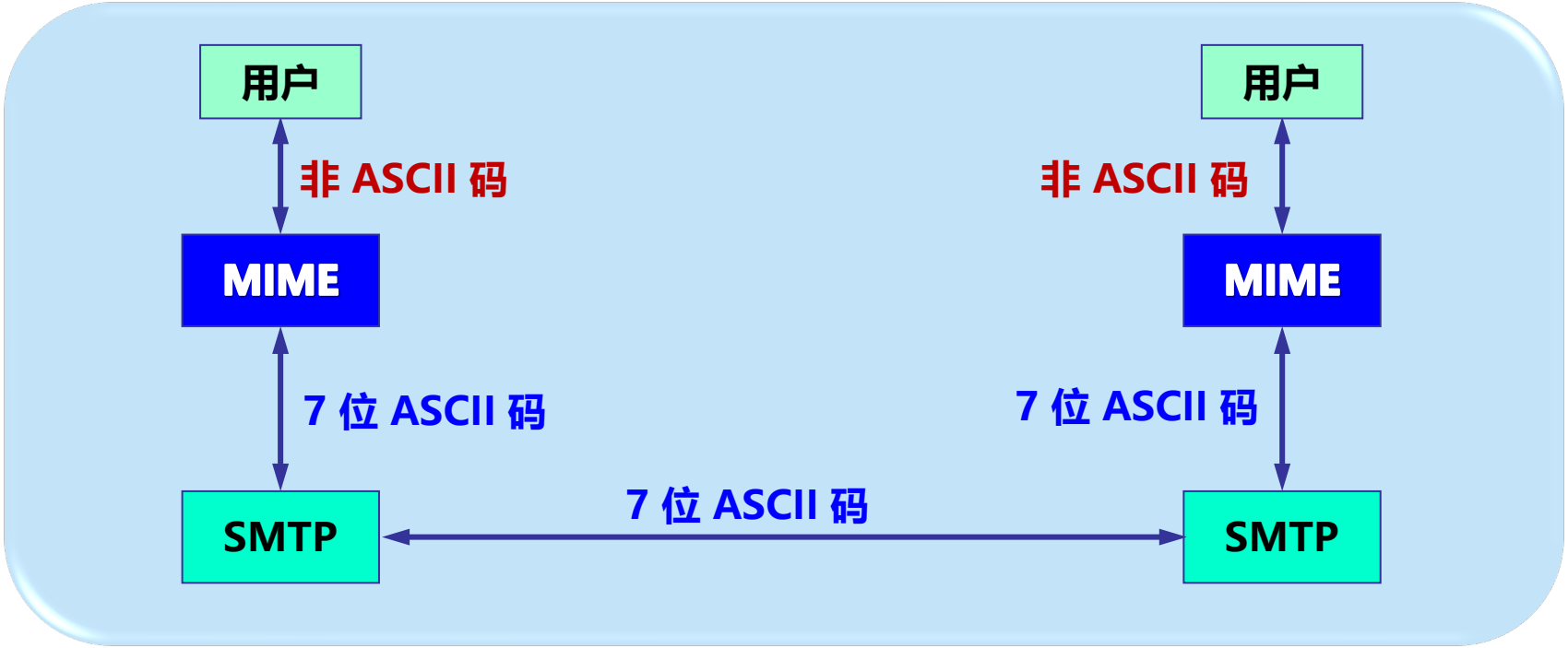
SMTP **缺点**：

- 不能传送可执行文件或其他的二进制对象。
- 限于传送 7 位的 ASCII 码，无法传送非 ASCII 编码的信息。
- 服务器会拒绝超过一定长度的邮件。
- 某些 SMTP 的实现并没有完全按照 [RFC 821] 的 SMTP 标准。

1. MIME 概述

- **通用互联网邮件扩充 MIME** 并没有改动 SMTP 或取代它。
- **意图：** 继续使用目前的 [RFC 822] 格式，但**增加**了邮件主体的结构，并定义了传送非 ASCII 码的**编码规则。**

MIME 和 SMTP 的关系



MIME 主要包括三个部分

- 5 个新的邮件首部字段。
- 定义了许多邮件内容的格式，对多媒体电子邮件的表示方法进行了标准化。
- 定义了传送编码，可对任何内容格式进行转换，而不会被邮件系统改变。

MIME 增加 5 个新的邮件首部

MIME 首部

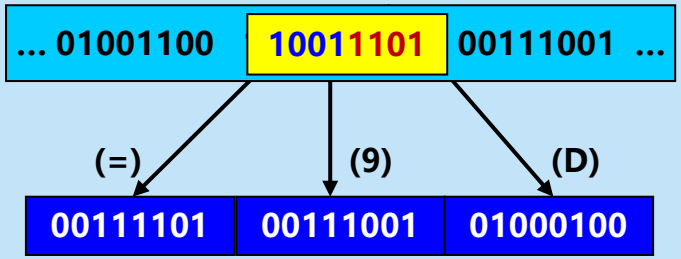
邮件首部
● MIME-Version：MIME 的版本。若无此行，则为英文文本。 ● Content-Description：这是可读字符串，此邮件的说明。 ● Content-Id：邮件的唯一标识符。 ● Content-Transfer-Encoding：传送时邮件主体使用的编码方法。 ● Content-Type：邮件内容类型 / 子类型。
邮件主体

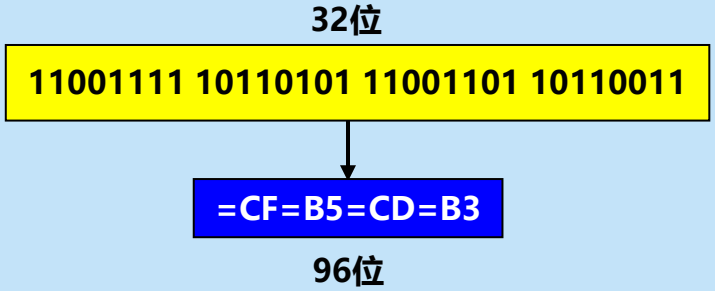
2. 内容传送编码(Content-Transfer-Encoding)

编码方法	说明
7bit	7 位 ASCII 编码，每行不能超过 1000 个字符（包括回车和换行）。默认编码方法。
8bit	8 位非 ASCII 编码，每行不能超过 1000 个字节（包括回车和换行）。
Binary	8 位非 ASCII 编码，任意长度的字节串。
Base64	将任意长度的字节串转换为用 7 位 ASCII 编码表示的字符串。可用于二进制和非文本数据的编码（任意的二进制文件）。
Quoted-printable	将任意长度的字节串转换为 ASCII 编码表示的字符串。可用于二进制和非文本数据的编码。（邮件中含有少量非ASCII码字符）

Quoted-printable 编码

- 所有可打印的ASCII码，除“=”外都不改变；
- 对于其他字符，先将每个字节的二进制代码用两个十六进制数字表示，然后在前面再加上一个等号“=”。
- “=”的二进制编码为“00111101”，十六进制为3D，所以它的编码为“=3D”。





原来 1 个字节，现在需要 3 个字节，开销 = 200%

Base64 编码

- 适合任意长度的二进制数据。
- 先把二进制代码划分成一个个24位长的单元；
- 然后每个24位单元划分为4个6位组；
- 每个6位组按以下方法转换成ASCII码。
  - 6位二进制代码共有64种不同的值：0-63；
  - 26个大写字母：A表示0，B表示1，等等；
  - 26个小写字母：a 表示26，B表示27，等等；
  - 10个数字：0表示52,1表示53，等等；
  - “+”表示62，“/”表示63。
  - “==”和“=”分别表示最后一组代码只有8位或16位。

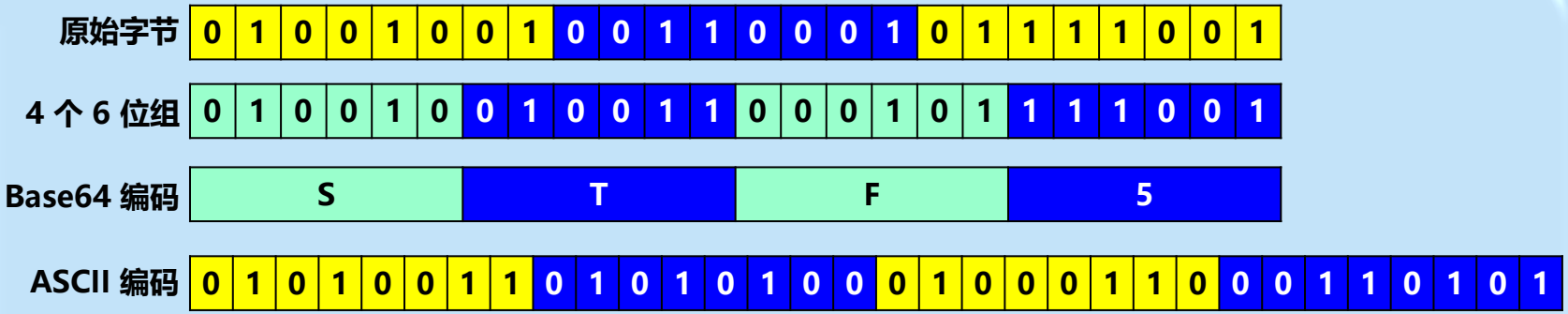
Base64 编码

- 编码表如下：

索引	对应字符	索引	对应字符	索引	对应字符	索引	对应字符	索引	对应字符
0	A	13	N	26	a	39	n	52	0
1	B	14	O	27	b	40	o	53	1
2	C	15	P	28	c	41	p	54	2
3	D	16	Q	29	d	42	q	55	3
4	E	17	R	30	e	43	r	56	4
5	F	18	S	31	f	44	s	57	5
6	G	19	T	32	g	45	t	58	6
7	H	20	U	33	h	46	u	59	7
8	I	21	V	34	i	47	v	60	8
9	J	22	W	35	j	48	w	61	9
10	K	23	X	36	k	49	x	62	+
11	L	24	Y	37	l	50	y	63	/
12	M	25	Z	38	m	51	z		

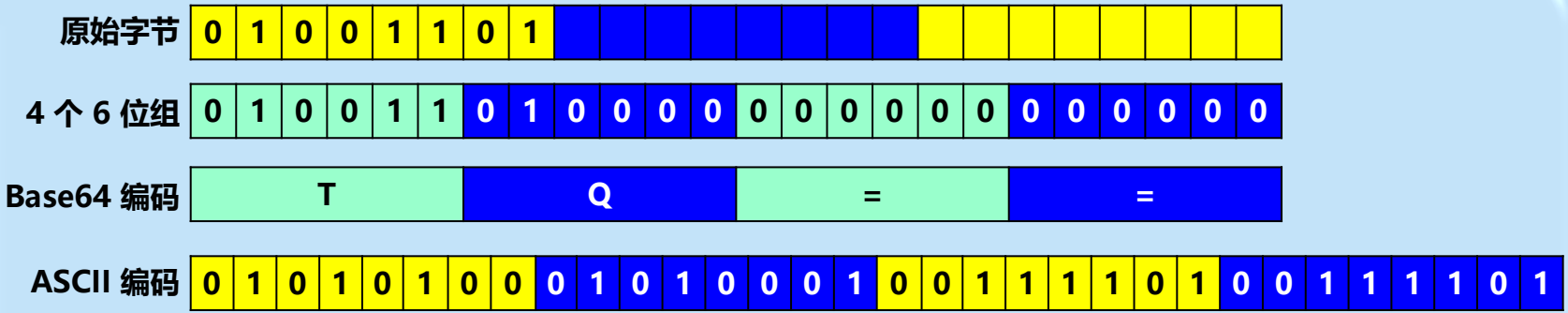
用两个连在一起的二个等号“==”和一个等号“=”分别表示最后一组的代码只有 8 位或 16 位。

Base64 编码



原来为 3 个字节，编码后为 4 个字节，开销 = 25%

Base64 编码


原始字节
0 1 0 0 1 1 0 1
4 个 6 位组
0 1 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Base64 编码
T
Q
=
=
ASCII 编码
0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1

3. 内容类型

- **MIME 标准规定：**

  Content-Type 说明必须含有两个标识符：内容类型 (type) 和子类型 (subtype)，中间用“/”分开。

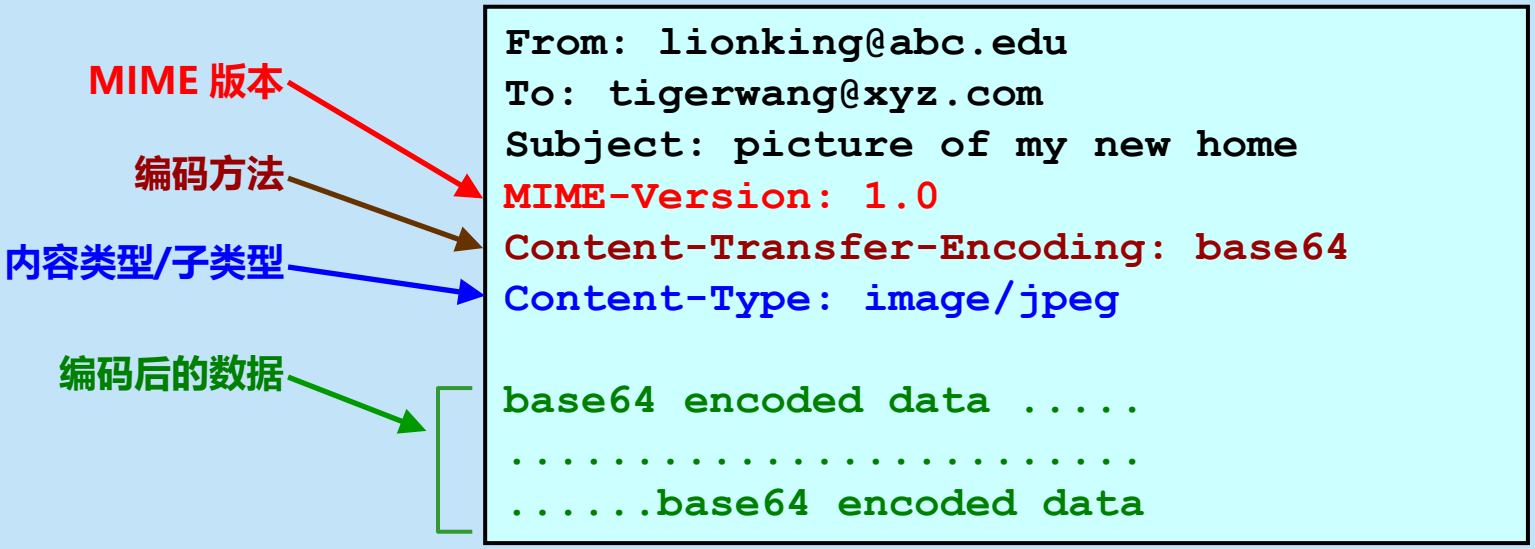
- **MIME 标准原先定义了 7 个基本内容类型和 15 种子类型。**
- **MIME 允许发件人和收件人自己定义专用的内容类型。但为避免可能出现名字冲突，标准要求为专用内容类型选择的名字要以字符串 X-开始。**

MIME Content-Type 说明中的类型及子类型

内容类型	子类型举例	说明
text（文本）	plain, html, xml, css	不同格式的文本
image（图像）	gif, jpeg, tiff	不同格式的静止图像
audio（音频）	basic, mpeg, mp4	可听见的声音
video（视频）	mpeg, mp4, quicktime	不同格式的影片
model（模型）	vrml	3D模型
application（应用）	octet-stream, pdf, javascript, zip	不同应用程序产生的数据
message（报文）	http, rfc822	封装的报文
multipart（多部分）	mixed, alternative, parallel, digest	多种类型的组合

Multipart 很有用，使邮件增加了相当大的灵活性。

MIME 举例


MIME 版本
编码方法
内容类型/子类型
编码后的数据
From: lionking@abc.edu
To: tigerwang@xyz.com
Subject: picture of my new home
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
base64 encoded data .....
.....
.....base64 encoded data

6.6 动态主机配置协议 DHCP

- 在协议软件中，给协议参数赋值的动作叫做**协议配置。**
- 一个协议软件在使用之前必须是已正确配置的。
- 具体的配置信息取决于协议栈。
- 连接到互联网的计算机的协议软件需要正确配置的参数包括：
  1. IP 地址
  2. 子网掩码
  3. 默认路由器的 IP 地址
  4. 域名服务器的 IP 地址

动态主机配置协议 DHCP

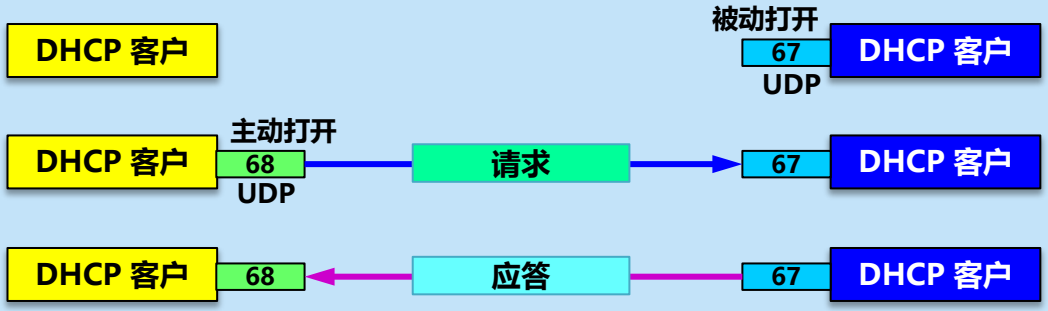
- 动态主机配置协议 DHCP (Dynamic Host Configuration Protocol) 提供了即插即用连网 (plug-and-play networking) 的机制，允许一台计算机加入网络和获取 IP 地址，而不用手工配置。
- DHCP 给运行服务器软件、且位置固定的计算机指派一个永久地址，给运行客户端软件的计算机分配一个临时地址。

DHCP 使用客户服务器方式

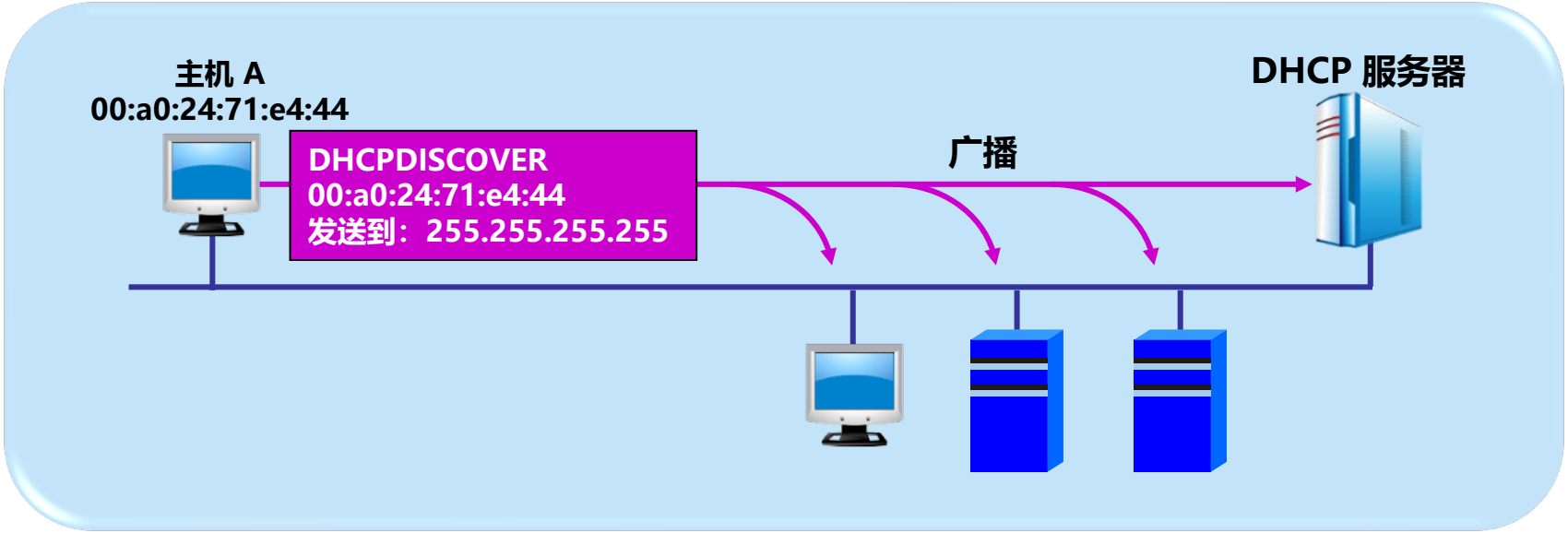
- 需要 IP 地址的主机在启动时就向 DHCP 服务器广播发送**发现报文（DHCPDISCOVER），**这时该主机就成为 DHCP 客户。
- 本地网络上所有主机都能收到此广播报文，但只有 DHCP 服务器才回答此广播报文。
- DHCP 服务器先在其数据库中查找该计算机的配置信息。若找到，则返回找到的信息。若找不到，则从服务器的 IP 地址池 (address pool) 中取一个地址分配给该计算机。DHCP服务器的回答报文叫做**提供报文（DHCPOFFER）**。

DHCP 工作方式

- DHCP 使用客户服务器方式，采用请求/应答方式工作。
- DHCP 基于 UDP 工作，DHCP 服务器运行在 67 号端口， DHCP 客户运行在 68 号端口。

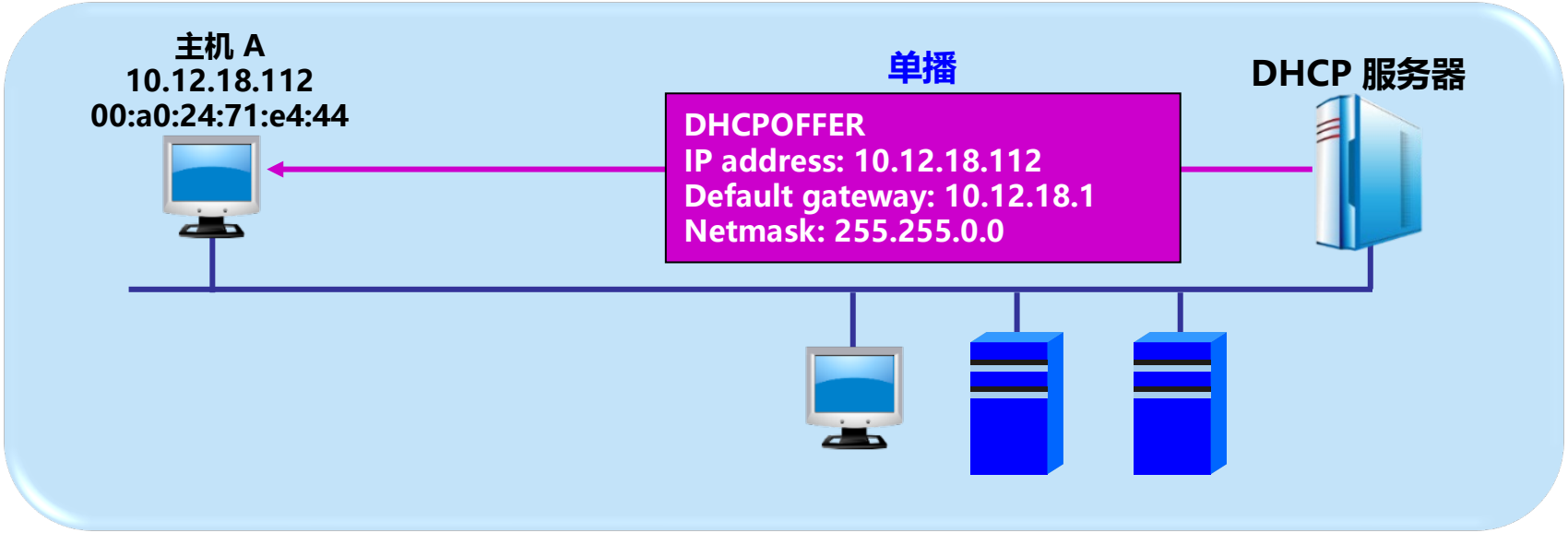


DHCP 工作方式



需要 IP 地址的主机向 DHCP 服务器广播发送发现报文 (DHCPDISCOVER) 。

DHCP 工作方式

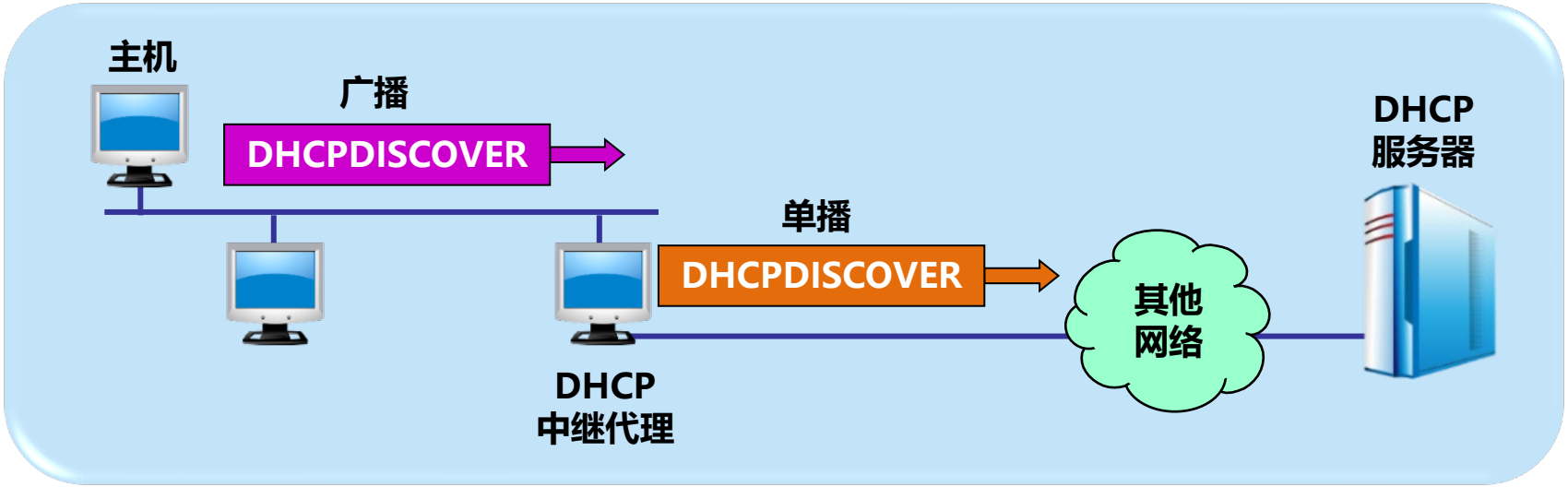


DHCP 服务器回答提供报文 (DHCPOFFER) （单播），提供 IP 地址等配置信息。

DHCP 中继代理 (relay agent)

- **问题：**每个网络上都需要有 DHCP 服务器吗?
- **答案：**不需要，因为会使 DHCP 服务器的数量太多。
- **问题：**若没有 DHCP 服务器，如何自动获得地址?
- **解决：**每一个网络**至少**有一个 DHCP **中继代理**，它配置了 DHCP 服务器的 IP 地址信息。

DHCP 中继代理以单播方式转发发现报文



- DHCP 中继代理收到主机广播发送的发现报文后，就以单播方式向 DHCP 服务器转发此报文，并等待其回答。
- 收到 DHCP 服务器回答的提供报文后，DHCP 中继代理再将其发回给主机。

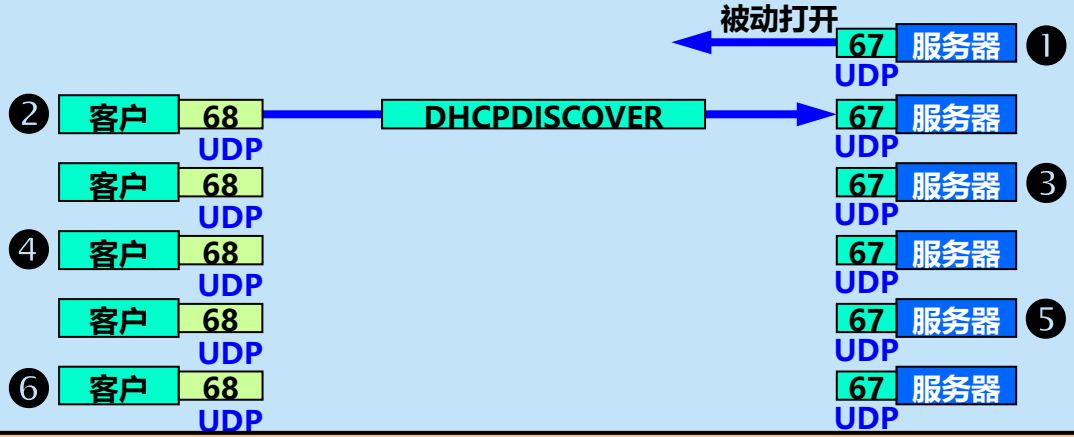
租用期 (lease period)

- DHCP 服务器分配给 DHCP 客户的 IP 地址的**临时的**，因此 DHCP 客户只能在一段有限的时间内使用这个分配到的 IP 地址。DHCP 协议称这段时间为**租用期。**
- 租用期的数值应由 DHCP 服务器自己决定。
- DHCP 客户也可在自己发送的报文中（例如，发现报文）提出对租用期的要求。

DHCP 协议的工作过程

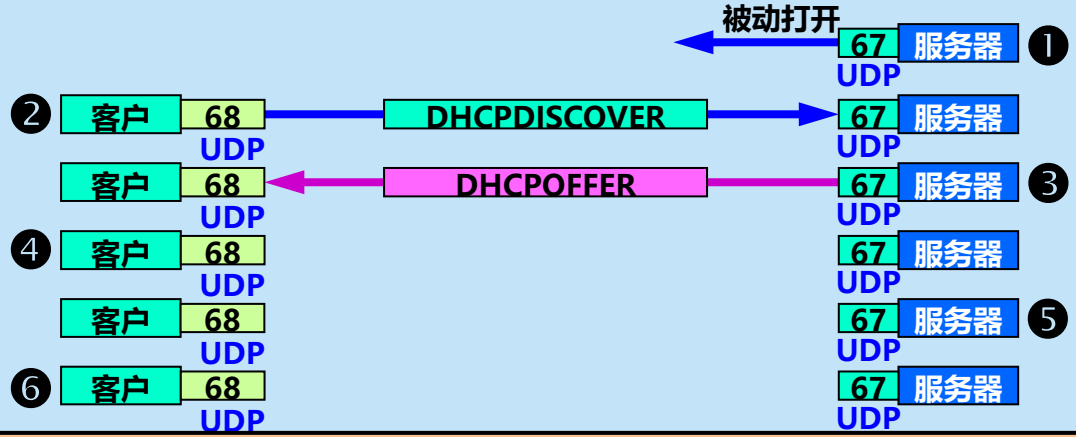
❶：DHCP 服务器**被动**打开 UDP 端口 67，等待客户端发来的报文。

DHCP 协议的工作过程



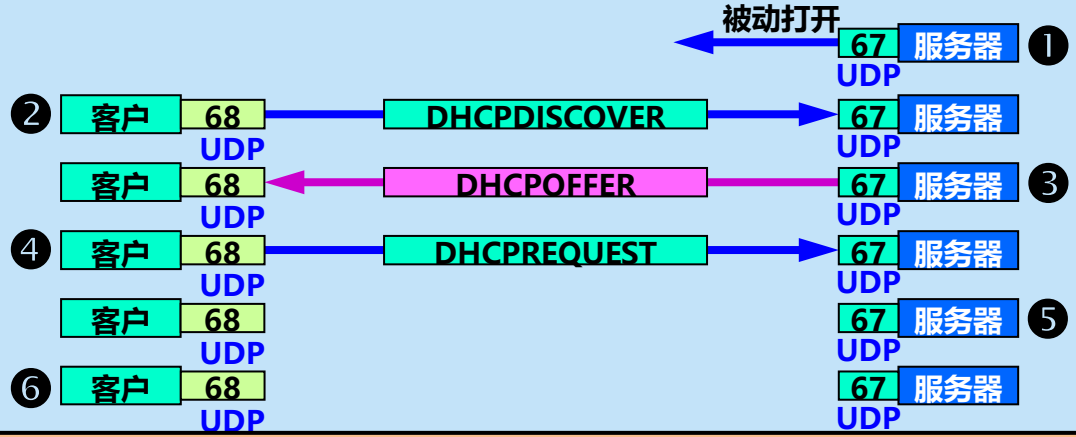
❷：DHCP 客户从 UDP 端口 68 发送 DHCP 发现报文 DHCPDISCOVER。

DHCP 协议的工作过程



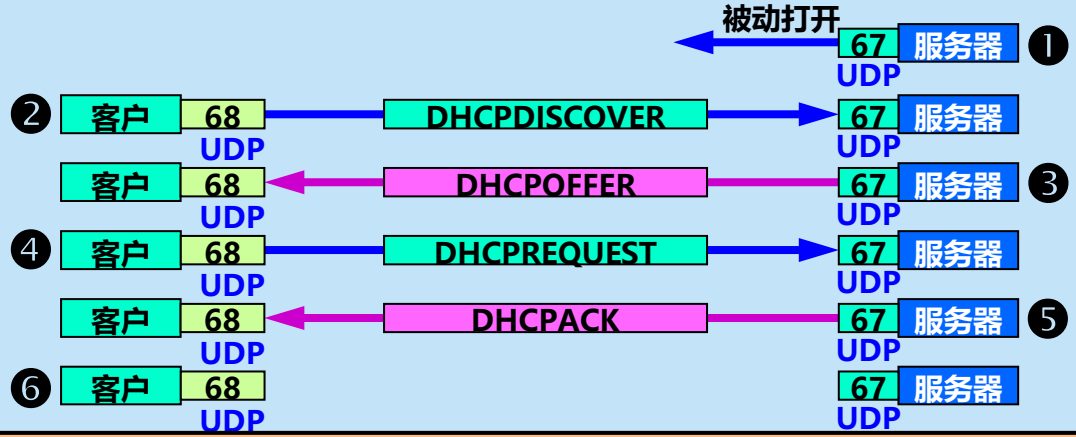
❸：凡收到 DHCP 发现报文的 DHCP 服务器**都发出** DHCP 提供报文 DHCPOFFER，因此 DHCP 客户可能收到多个 DHCP 提供报文 。

DHCP 协议的工作过程



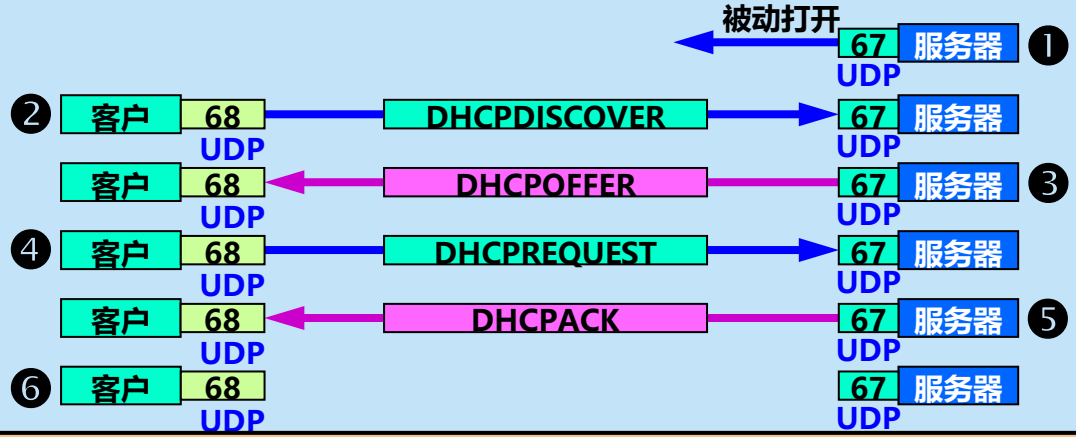
❹：DHCP 客户从几个 DHCP 服务器中选择其中的一个，并向所选择的 DHCP 服务器发送 DHCP 请求报文 DHCPREQUEST。

DHCP 协议的工作过程



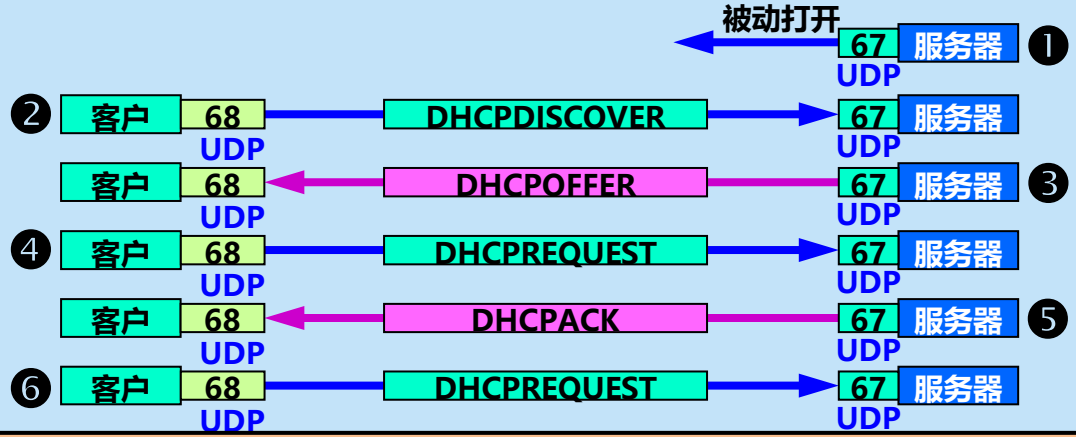
❺：被选择的 DHCP 服务器发送确认报文 DHCPACK，DHCP 客户可开始使用得到的临时 IP 地址了，进入已绑定状态。

DHCP 协议的工作过程



DHCP 客户现在要根据服务器提供的**租用期 T** 设置**两个**计时器 T_1 和 T_2，它们的超时时间分别是 0.5T 和 0.875T。当超时时间到时，就要请求更新租用期。

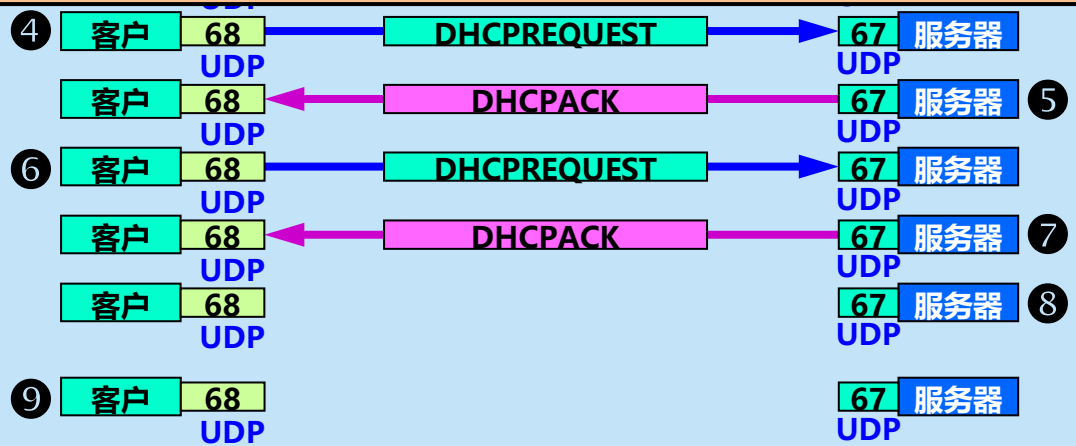
DHCP 协议的工作过程



❻：租用期过了**一半**（T_1 时间到），DHCP 发送请求报文 DHCPREQUEST，要求**更新**租用期。

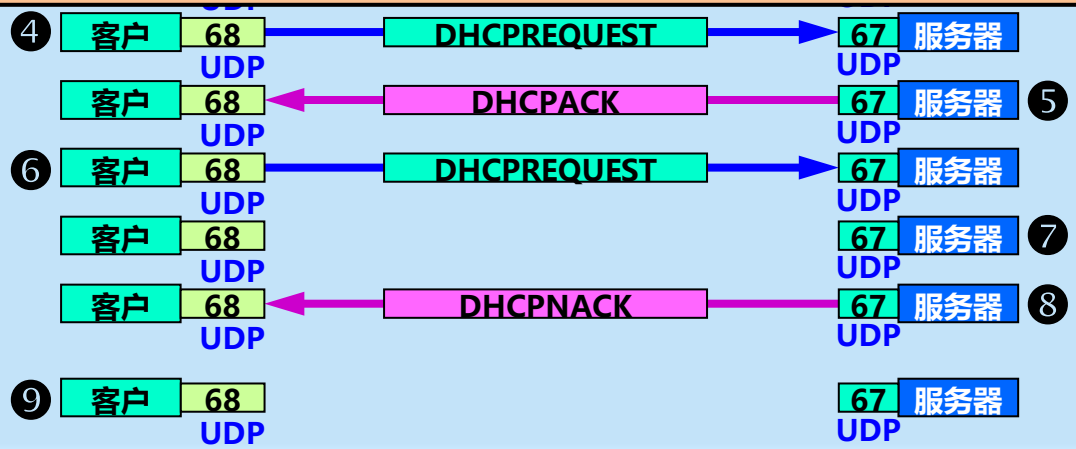
DHCP 协议的工作过程

❼：DHCP 服务器若同意，则发回确认报文 DHCPACK。DHCP 客户得到了新的租用期，重新设置计时器。



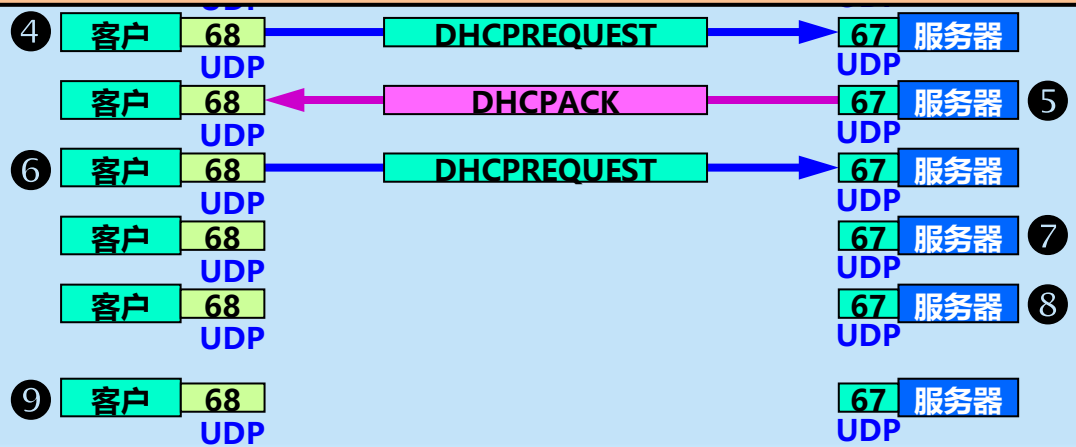
DHCP 协议的工作过程

❽：DHCP 服务器若不同意，则发回否认报 DHCPNACK。这时 DHCP 客户必须立即停止使用原来的 IP 地址，而必须重新申请 IP 地址（回到步骤 ❷）。



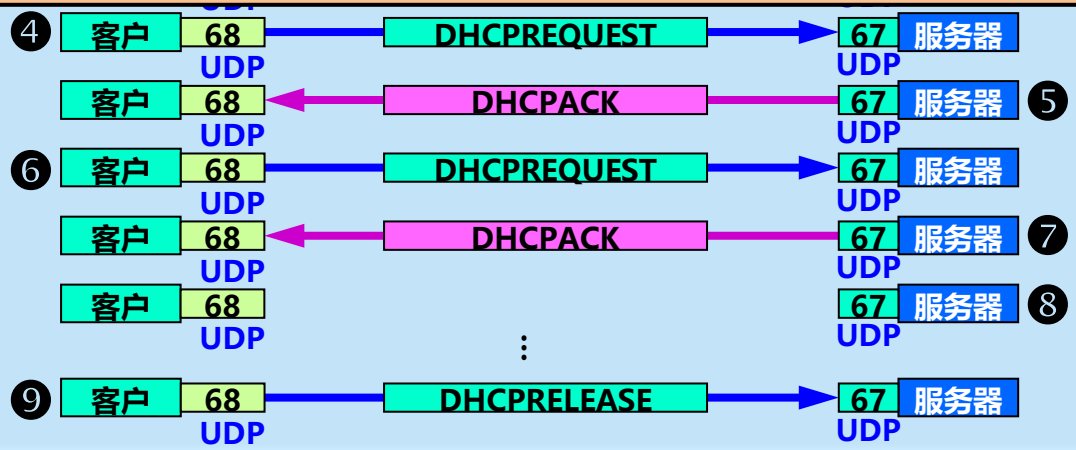
DHCP 协议的工作过程

若 DHCP 服务器不响应步骤 ❻ 的请求报文 DHCPREQUEST，则在租用期过了 87.5% 时 (T_2 时间到)，DHCP 客户必须重新发送请求报文 DHCPREQUEST（重复步骤 ❻），然后又继续后面的步骤。



DHCP 协议的工作过程

❾：DHCP 客户可随时提前终止服务器所提供的租用期，这时只需向 DHCP 服务器发送释放报文 DHCPRELEASE 即可。



6.7 简单网络管理协议 SNMP

6.7.1 网络管理的基本概念

6.7.2 管理信息结构 SMI

6.7.3 管理信息库 MIB

6.7.4 SNMP 的协议数据单元和报文

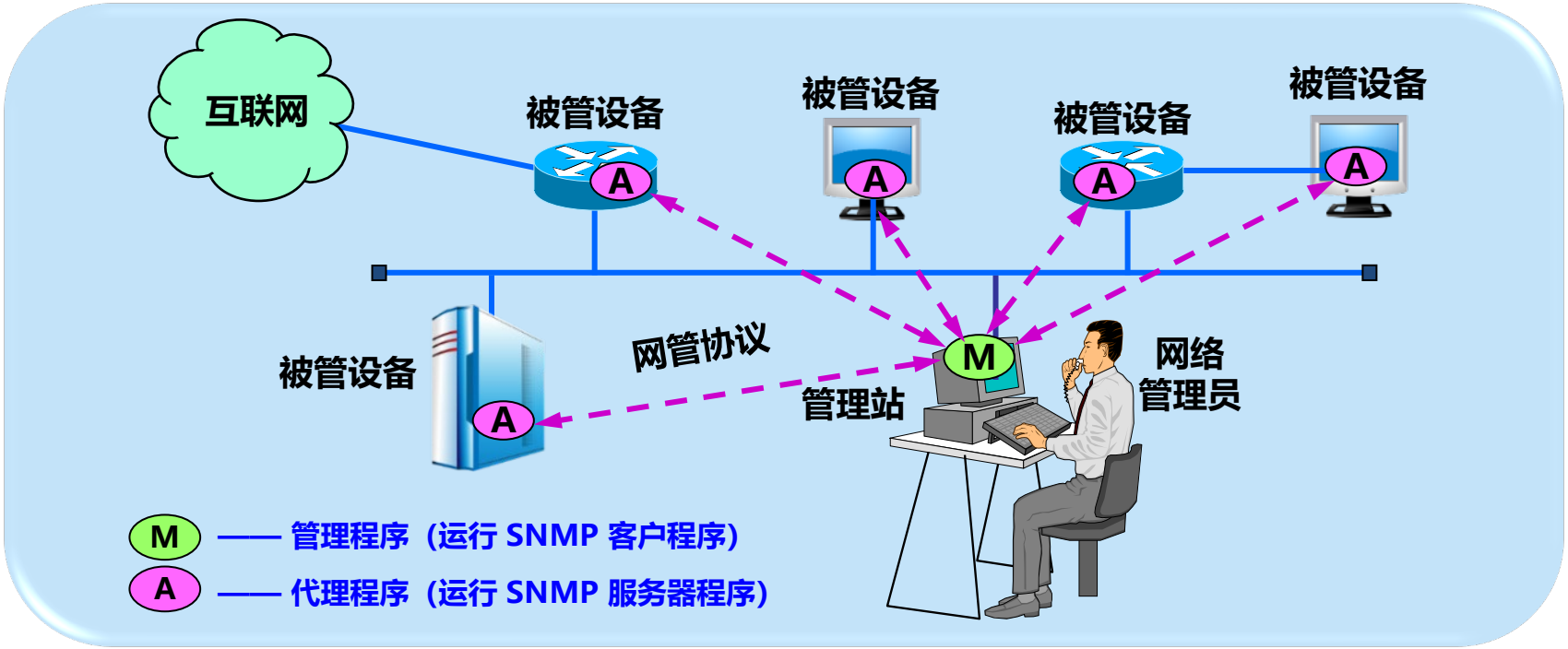
6.7.1 网络管理的基本概念

- **网络管理**包括对硬件、软件和人力的使用、综合与协调，以便对网络资源进行监视、测试、配置、分析、评价和控制，这样就能以合理的价格满足网络的一些需求，如实时运行性能，服务质量等。
- 网络管理常简称为**网管**。

网络管理的五大功能

- **故障管理**：故障检测、隔离和纠正。
- **配置管理**：初始化网络、并配置网络。
- **计费管理**：记录网络资源的使用。
- **性能管理**：估价系统资源的运行状况及通信效率等。
- **网络安全管理**：对授权机制、访问控制、加密和加密关键字的管理。

网络管理的一般模型


互联网
被管设备
被管设备
被管设备
被管设备
被管设备
网管协议
管理站
网络
管理员
M —— 管理程序（运行 SNMP 客户程序）
A —— 代理程序（运行 SNMP 服务器程序）

网络管理模型中的主要构件

- 管理站也常称为网络运行中心 NOC (Network Operations Center)，是网络管理系统的核心。
- 管理程序是管理站中的关键构件，在运行时就成为管理进程。
- 管理站（硬件）或管理程序（软件）都可称为管理者(manager)。Manager 不是指人，而是指机器或软件。
- 网络管理员 (administrator) 指的是负责网络管理的人员。
- 大型网络往往实行多级管理，因而有多个管理者，而一个管理者一般只管理本地网络的设备。

被管对象 (Managed Object)

- 网络的每一个被管设备（包括设备中的软件）中可能有多个**被管对象**。
- 被管设备有时可称为**网络元素**或**网元**。
- 在被管设备中也会有一些不能被管的对象。

代理 (agent)

- 在每一个被管设备中都要运行一个程序，以便和管理站中的管理程序进行通信。
- 这些运行着的程序叫做网络管理代理程序，或简称为代理。
- 代理程序在管理程序的命令和控制下在被管设备上采取本地的行动。

网络管理协议

- **网络管理协议**简称为**网管协议。**
- 网络管理协议是管理程序和代理程序之间进行通信的规则。
- 网络管理员利用网络管理协议，通过管理站对网络中的被管设备进行管理。
- **注意：网管协议本身不管理网络。**

简单网络管理协议 SNMP

- 简单网络管理协议 SNMP (Simple Network Management Protocol) 中的管理程序和代理程序按客户服务器方式工作。
- 管理程序运行 SNMP 客户程序，向某个代理程序发出请求 (或命令)。
- 代理程序运行 SNMP 服务器程序，返回响应 (或执行某个动作)。
- 在网管系统中，往往是一个（或少数几个）客户程序与很多的服务器程序进行交互。

网络管理的基本原理

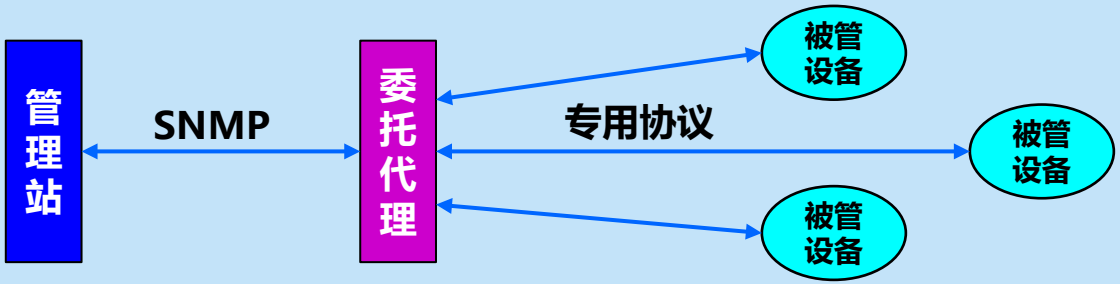
若要管理某个对象，就必然会给该对象添加一些软件或硬件，但这种“添加”必须对原有对象的影响尽量小些。

SNMP 的基本功能

- 最重要的指导思想：**尽可能简单。**
- 基本功能：
  1. 监视网络性能
  2. 检测分析网络差错
  3. 配置网络设备等。

SNMP 的管理站和委托代理

- 整个系统必须有一个**管理站**。
- 若网络元素使用的不是 SNMP 而是另一种网络管理协议，SNMP 协议就无法控制该网络元素。这时可使用**委托代理** (proxy agent)。
- 委托代理能提供协议转换和过滤操作等功能，对被管对象进行管理。



SNMP 网络管理组成

- SNMP 的网络管理由三个部分组成：
  1. SNMP 本身
  2. 管理信息结构 SMI (Structure of Management Information)
  3. 管理信息库 MIB (Management Information Base)。

简单网络管理协议 SNMP

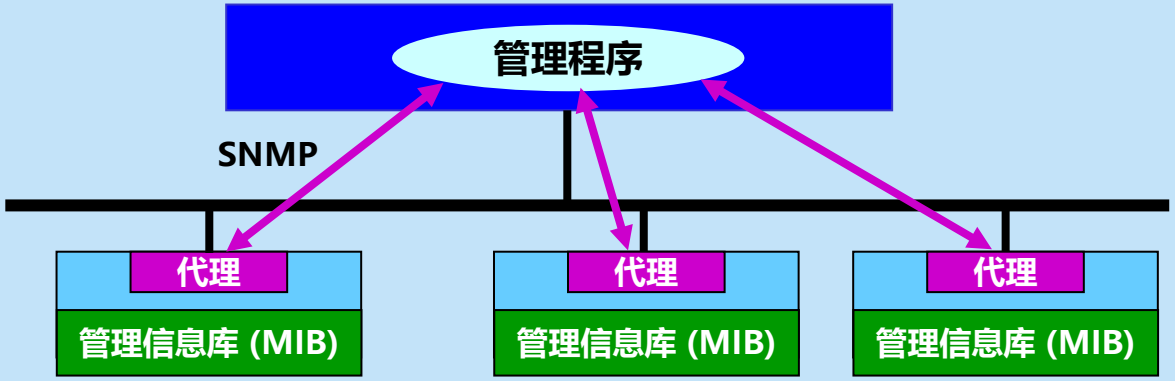
- SNMP 定义了管理站和代理之间所交换的**分组格式**。
- 所交换的分组包含各代理中的对象（**变量**）名及其状态（**值**）。
- SNMP 负责读取和改变这些数值。

管理信息结构 SMI

- SMI 定义了命名对象和定义对象类型（包括范围和长度）的**通用规则**，以及把对象和对象的值进行**编码的规则**，以确保网络管理数据的语法和语义的**无二义性**。
- 但从 SMI 的名称并不能看出它的功能。
- SMI 并不定义一个实体应管理的对象数目，也不定义被管对象名以及对象名及其值之间的关联。

管理信息库 MIB

- MIB 在被管理的实体中创建了命名对象，并规定了其类型。
- 管理程序使用 MIB 中的信息，对网络进行管理。

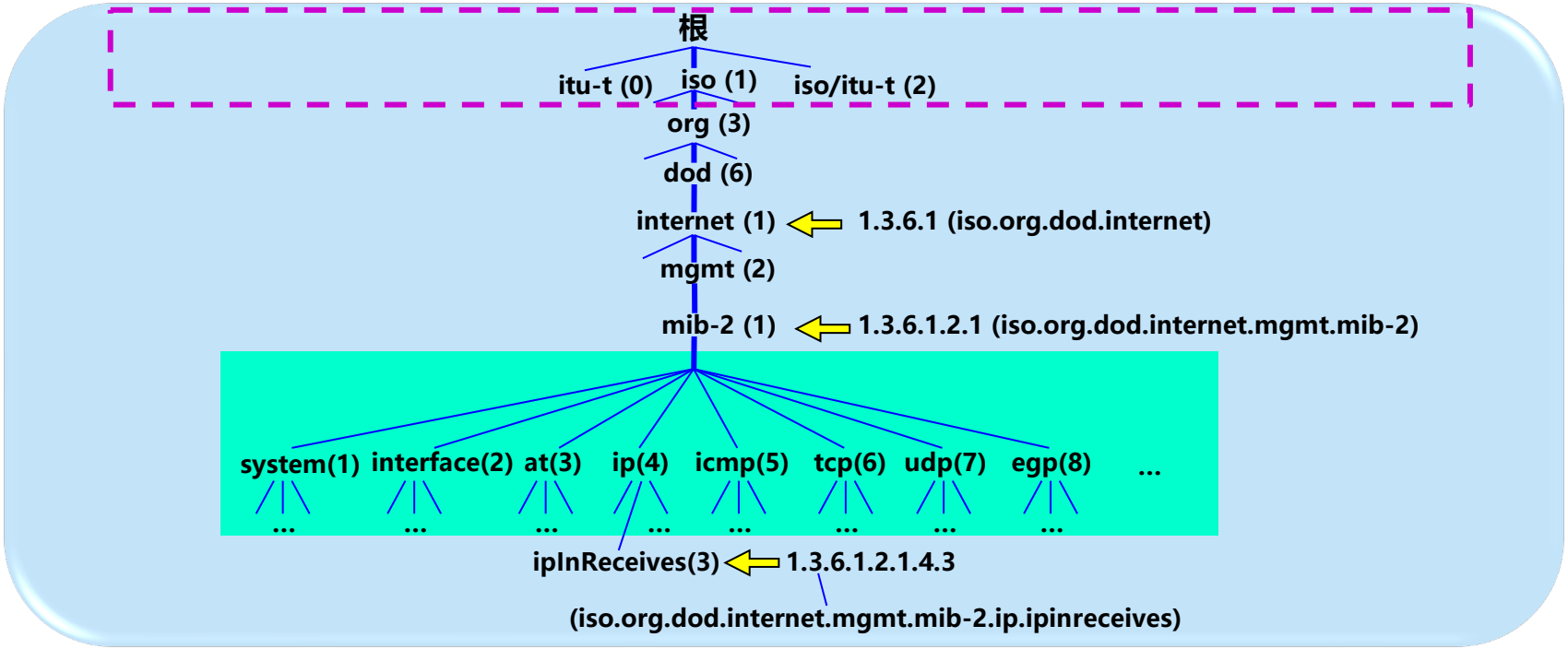


6.7.2 管理信息结构 SMI

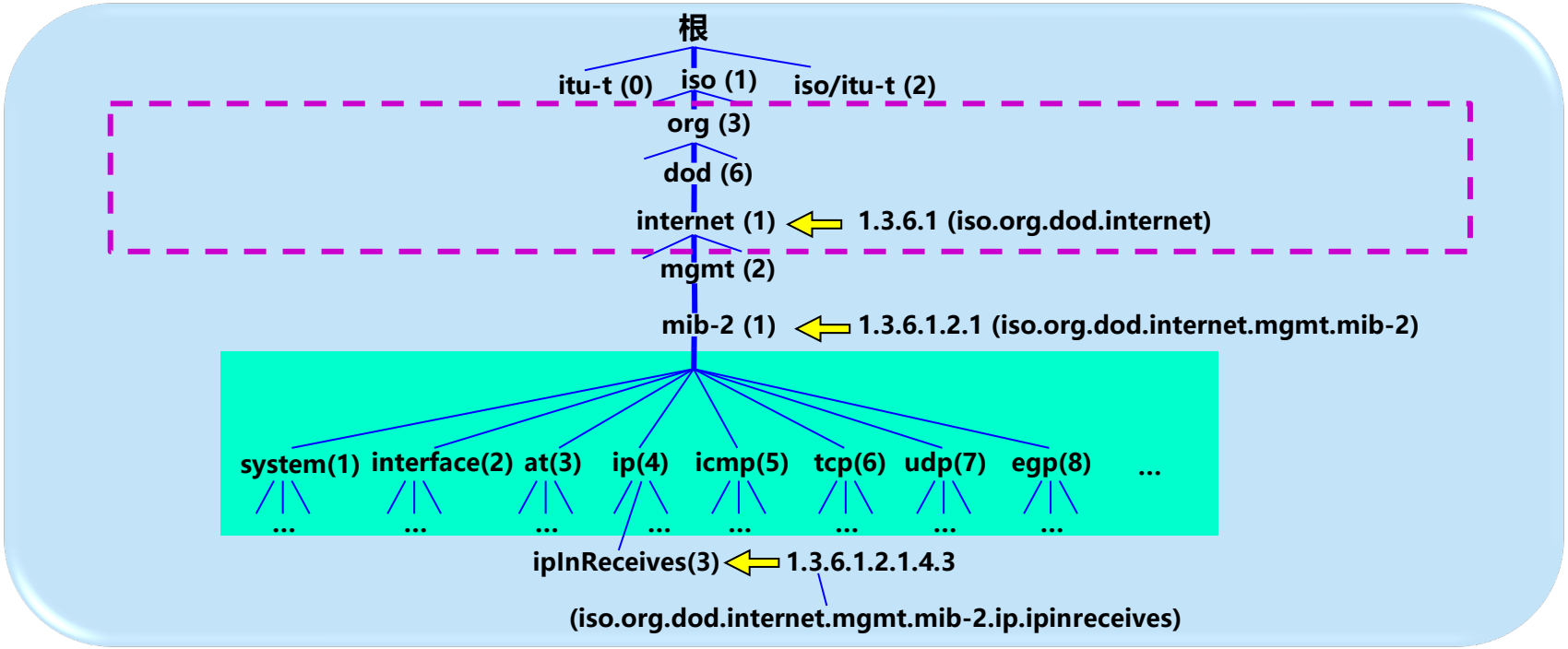
SMI 的功能：

1. **被管对象应怎样命名；**
2. **用来存储被管对象的数据类型有哪些种；**
3. **在网络上传送的管理数据应如何编码。**

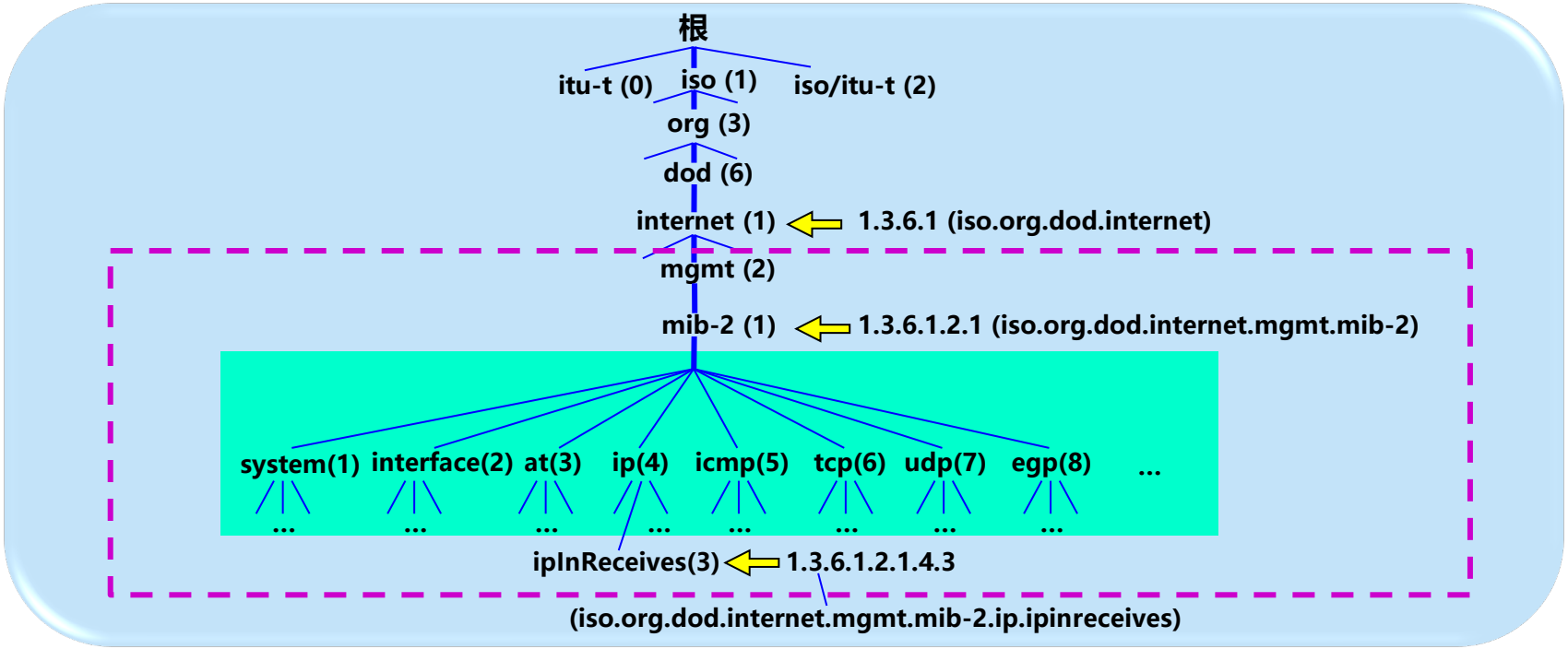
SMI 规定：所有被管对象必须在命名树上



SMI 规定：所有被管对象必须在命名树上



SMI 规定：所有被管对象必须在命名树上



SMI 使用 ASN.1

- SMI 标准指明：所有的 MIB 变量必须使用抽象语法记法 1 (ASN.1) 来定义。
- SMI 既是 ASN.1 的子集，又是 ASN.1 的超集。
- ASN.1 的记法很严格，使得数据的含义不存在任何可能的二义性。

SMI 数据类型

数据
- 类型：值集合的名字
- 值：某个值集合中的一个元素

- SMI 把数据类型分为两大类：
  1. 简单类型
  2. 结构化类型。

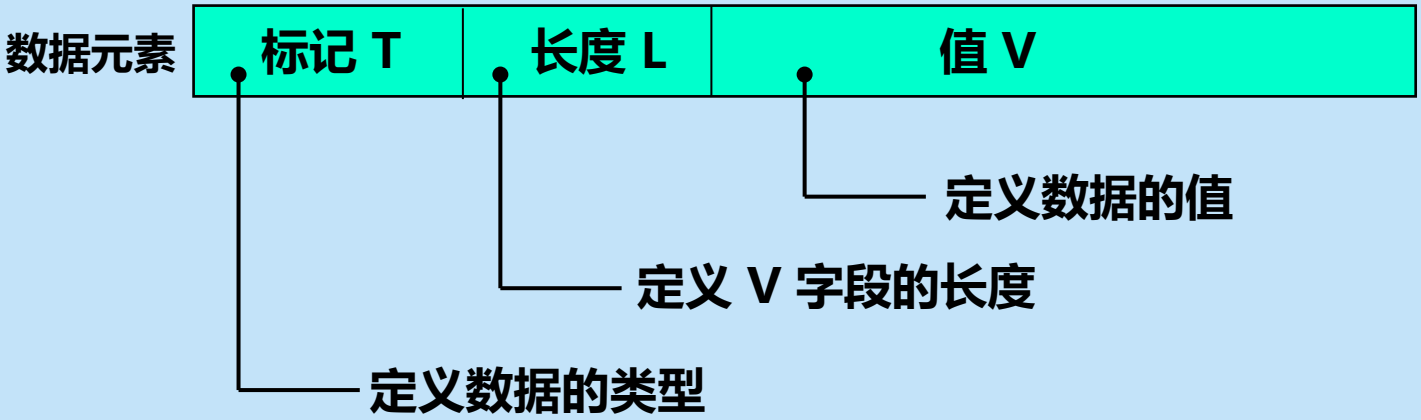
ASN.1 部分数据类型

分类	数据类型名称	含义
简单类型	INTEGER	任意长度整数
	BIT STRING	0 位或多位组成的二进制串
	OCTET STRING	0 位或多位组成的字节串
	NULL	空值
	OBJECT IDENTIFIER	定义的数据类型
结构化类型	SEQUENCE	由多个数据类型按序组成的值
	SEQUENCE OF	由同一数据类型按序组成的值
	CHOICE	可以从多个数据类型中选择一个
	ANY	任何数据类型

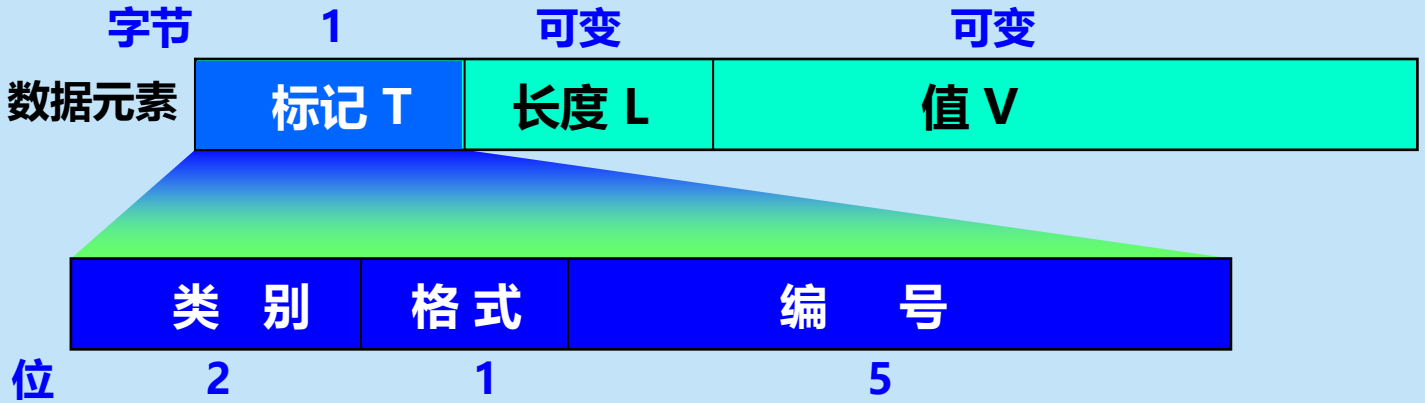
基本编码规则 BER

- ISO 在制订 ASN.1 语言的同时也为它定义了一种标准的编码方案，即基本编码规则 BER (Basic Encoding Rule)。
- BER 指明了每种数据类型中每个数据的值的表示。
- SMI 使用 ASN.1 制定的 BER 进行数据的编码。
- 发送端用 BER 编码，可将用 ASN.1 所表述的报文转换成唯一的比特序列。
- 接收端用 BER 进行解码，得到该比特序列所表示的 ASN.1 报文。

用 TLV 方法进行编码



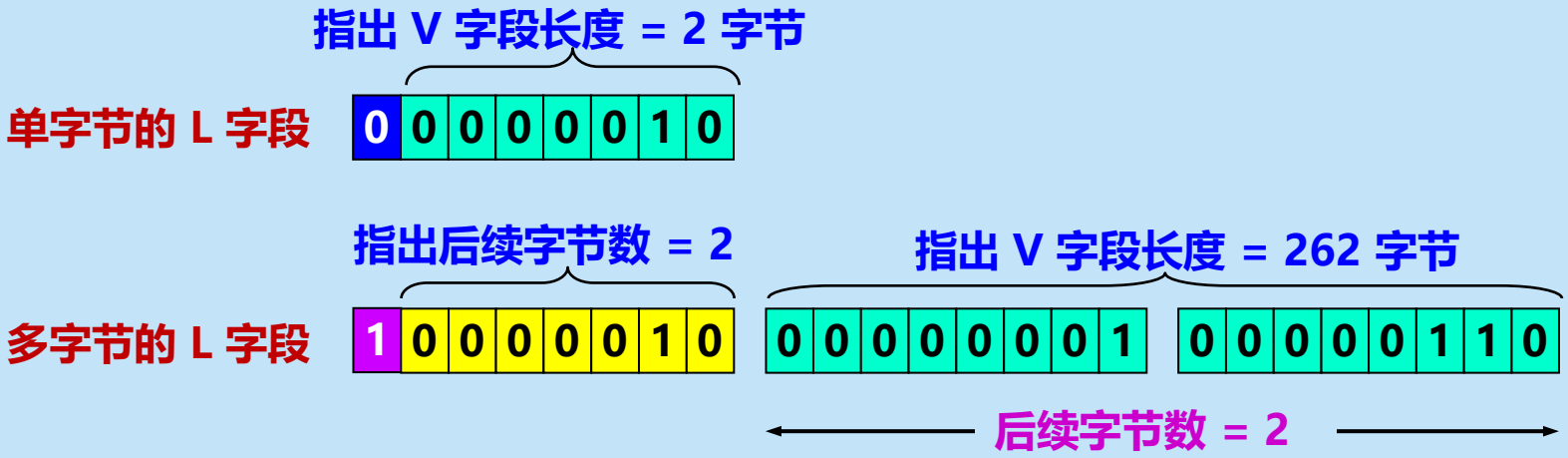
TLV 中的 T 字段定义数据的类型



TLV 中的 T 字段定义数据的类型

数据类型	类别	格式	编号	T 字段（二进制）	T 字段（十六进制）
INTEGER	00	0	00010	00000010	02
OCTET STRING	00	0	00100	00000100	04
OBJECT IDENTIFIER	00	0	00110	00000110	06
NULL	00	0	00101	00000101	05
Sequence, sequence of	00	1	10000	00110000	30
IPAddress	01	0	00000	01000000	40
Counter	01	0	00001	01000001	41
Gauge	01	0	00010	01000010	42
TimeTicks	01	0	00011	01000011	43
Opaque	01	0	00100	01000100	44

TLV 中的 L 字段定义 V 字段的长度



TLV 中的 V 字段定义数据的值

类　型	大 小	说　　明
INTEGER	4 字节	在 -2^{31} 到 $2^{31}-1$ 之间的整数
Interger32	4 字节	和 INTEGER 相同
Unsigned32	4 字节	在 0 到 $2^{32}-1$ 之间的无符号数
OCTET STRING	可变	不超过 65535 字节长的字节串
OBJECT IDENTIFIER	可变	对象标识符
IPAddress	4 字节	由 4 个整数组成的IP地址
Counter32	4 字节	可从 0 增加到 2^{32} 的整数；当它到达最大值时就返回到 0
TimeTicks	4 字节	记录时间的计数值，以 1/100 秒为单位
BITS	—	比特串
Opaque	可变	不解释的串

TLV 中的 V 字段定义数据的值

- 例如，INTEGER 15，其 T 字段是 02， INTEGER 类型要用 4 字节编码。最后得出 TLV 编码为 02 04 00 00 00 0F。

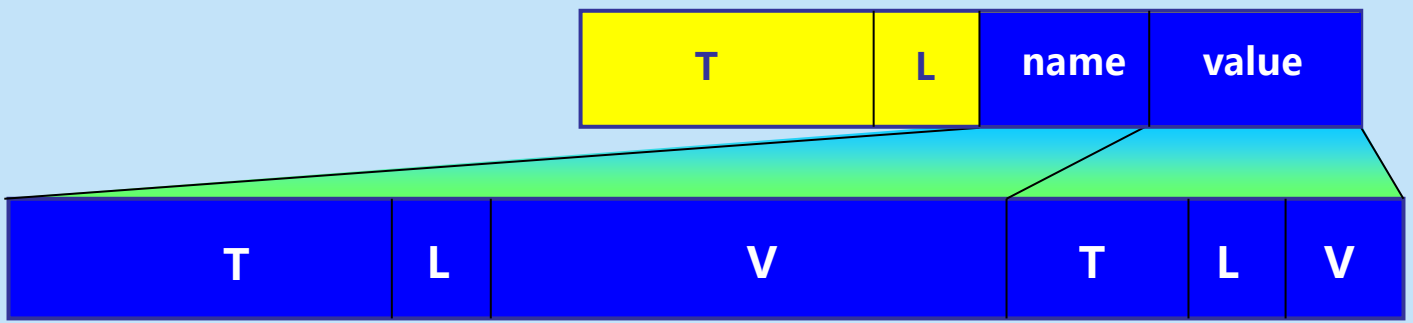
T	L	V
02	04	00 00 00 0F

TLV 中的 V 字段定义数据的值

- 例如，IPAddress = 192.1.2.3，其 T 字段是 40，V 字段需要 4 字节表示，因此得出 TLV 编码是 40 04 C0 01 02 03。

T 40	L 04	V C0 01 02 03

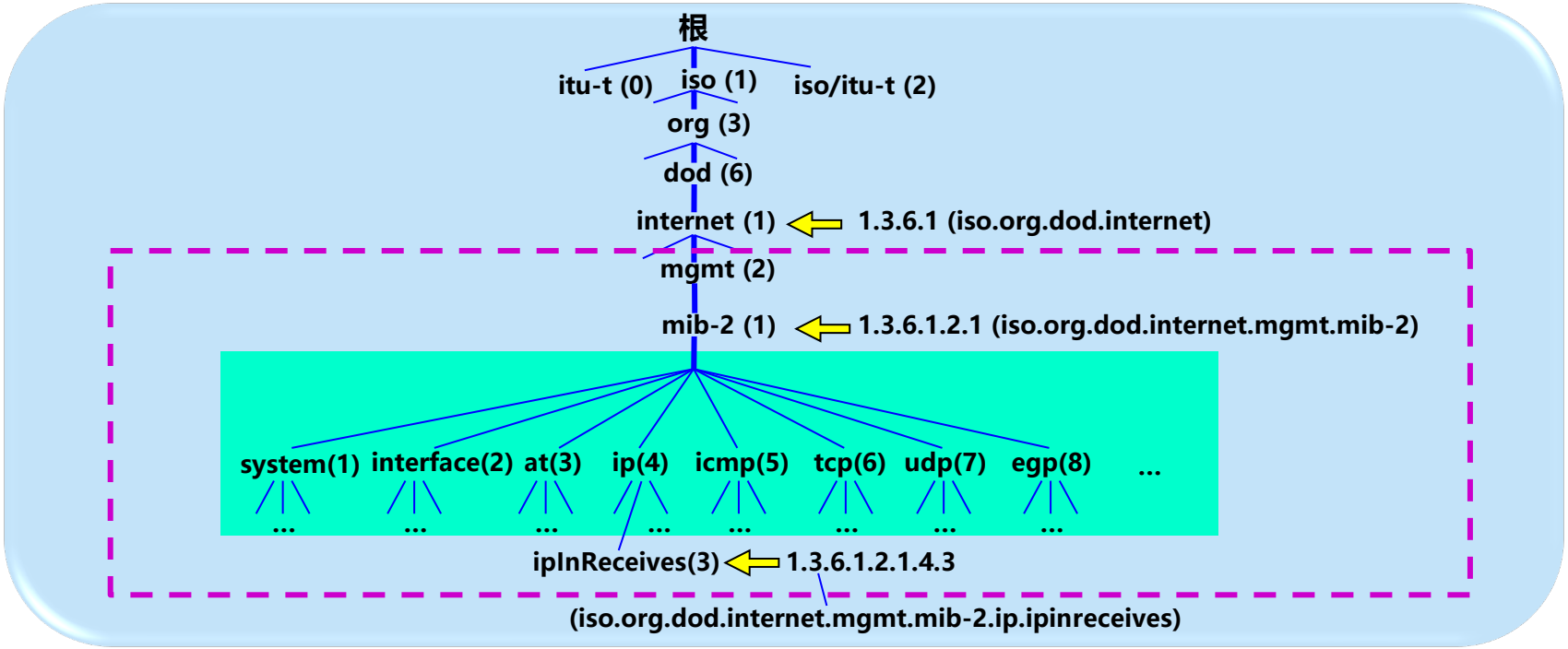
TLV 中的 V 字段可嵌套其他数据元素的 TLV 字段



6.7.3 管理信息库 MIB

- 被管对象必须维持可供管理程序读写的若干控制和状态信息。这些信息总称为**管理信息库** MIB (Management Information Base)。
- 管理程序使用 MIB 中这些信息的**值**对网络进行管理（如读取或重新设置这些值）。
- 只有在 MIB 中的对象才是 SNMP 所能够管理的。

SMI 规定：所有被管对象必须在命名树上



节点 mib-2 所包含的信息类别举例

类　别	标　号	所包含的信息
system	(1)	主机或路由器的操作系统
interfaces	(2)	各种网络接口
address translation	(3)	地址转换（例如，ARP 映射）
ip	(4)	IP 软件
icmp	(5)	ICMP 软件
tcp	(6)	TCP 软件
udp	(7)	UDP 软件
egp	(8)	EGP 软件

MIB 变量的例子

MIB 变量	所属类别	意　义
sysUpTime	system	距上次重启动的时间
ifNumber	interfaces	网络接口数
ifMtu	interfaces	特定接口的最大传送单元 MTU
ipDefaultTTL	ip	IP 在生存时间字段中使用的值
ipInReceives	ip	接收到的数据报数目
ipForwDatagrams	ip	转发的数据报数目
ipOutNoRoutes	ip	路由选择失败的数目
ipReasmOKs	ip	重装的数据报数目
ipFragOKs	ip	分片的数据报数目
ipRoutingTable	ip	IP 路由表
icmpInEchos	icmp	收到的 ICMP 回送请求数目
tcpRtoMin	tcp	TCP 允许的最小重传时间
tcpMaxConn	tcp	允许的最大 TCP 连接数目
tcpInSegs	tcp	已收到的 TCP 报文段数目
udpInDatagrams	udp	已收到的 UDP 数据报数目

6.7.4 SNMP 的协议数据单元和报文

- SNMP 的操作只有**两种**基本的管理功能：
  1. “**读**”操作，用 get 报文来检测各被管对象的状况；
  2. “**写**”操作，用 set 报文来改变各被管对象的状况。
- SNMP 的这些功能通过**探询**操作来实现。

SNMP 的探询操作

- SNMP 管理进程**定时**向被管理设备**周期性**地发送探询信息。
- **好处：**
  1. 可使系统相对简单。
  2. 能限制通过网络所产生的管理信息的通信量。
- **缺点：**
  1. 不够灵活，而且所能管理的设备数目不能太多。
  2. 开销也较大。

陷阱 (trap)

- SNMP 允许不经过询问就能发送某些信息。这种信息称为陷阱，表示它能够捕捉“事件”。
- 当被管对象的代理检测到有事件发生时，就检查其门限值。代理只向管理进程报告达到某些门限值的事件（即过滤）。
- 过滤的好处：
  1. 仅在严重事件发生时才发送陷阱；
  2. 陷阱信息很简单且所需字节数很少。

SNMP 是有效的网络管理协议

- 使用**探询**（至少是周期性地）以维持对网络资源的实时监视。
- 同时也采用**陷阱**机制报告特殊事件，使得 SNMP 成为一种有效的网络管理协议。

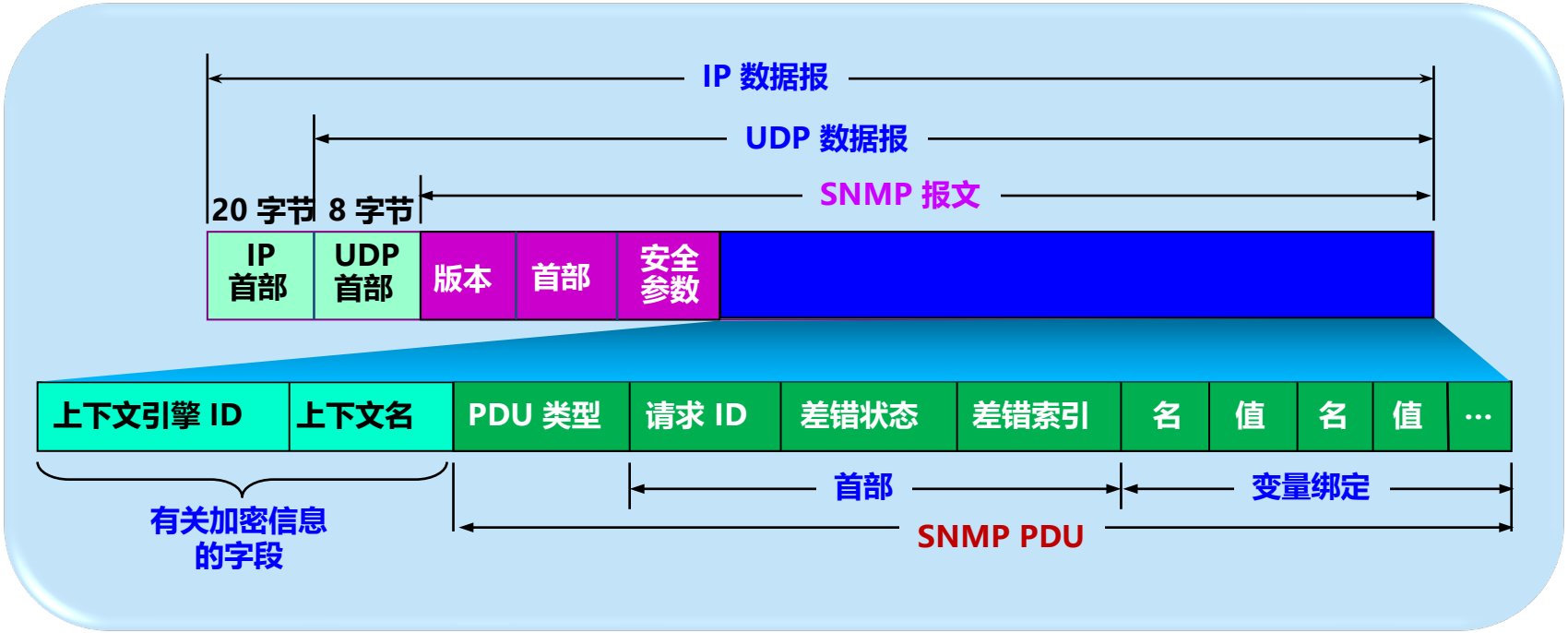
SNMP 使用无连接的 UDP

- 运行代理程序的服务器端用 UDP 熟知端口 161 接收 get 或 set 报文，发送响应报文。与熟知端口通信的客户端使用临时端口。
- 运行管理程序的客户端则使用 UDP 熟知端口 162 来接收来自各代理的 trap 报文。

SNMPv1 定义的协议数据单元 (PDU) 类型

PDU 编号 (T字段)	PDU 名称	用途
0 (A0)	GetRequest	用来查询一个或一组变量的值
1 (A1)	GetNextRequest	允许在 MIB 树上读取下一个变量，此操作可反复进行
2 (A2)	Reponse	代理向管理者或管理者向管理者发送响应
3 (A3)	SetRequest	对一个或多个变量值进行设置
5 (A5)	GetBulkRequest	管理者从代理读取大数据块的值
6 (A6)	InformRequest	管理者从另一管理者读取代理的变量
7 (A7)	SNMPv2Trap	代理向管理者报告异常事件
8 (A8)	Report	管理者之间报告某些差错

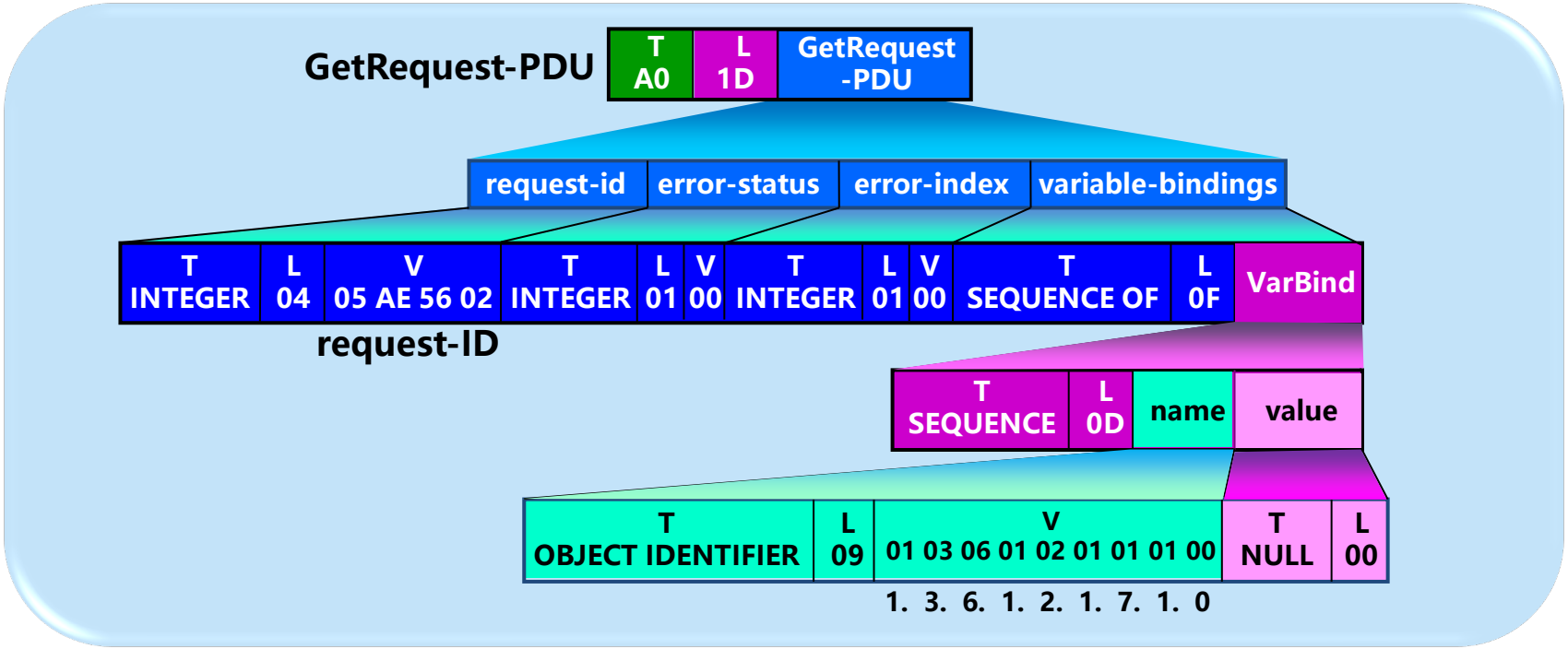
SNMP 的报文格式


IP 数据报
UDP 数据报
SNMP 报文
20 字节
8 字节
IP 首部
UDP 首部
版本
首部
安全参数
上下文引擎 ID
上下文名
PDU 类型
请求 ID
差错状态
差错索引
名
值
名
值
…
首部
变量绑定
有关加密信息的字段
SNMP PDU

Get-request 报文 ASN.1 定义

```
Get-request-PDU ::= [0]                    --[0] 表示上下文类，编号为 0
  IMPLICIT SEQUENCE {                      --类型是 SEQUENCE
    request-id       integer32,            --变量 request-id 的类型是 integer32
    error-status     INTEGER {0..18},      --变量 error-status 取值为 0 ~ 18 的整数
    error-index      INTEGER {0..max-bindings}, --变量 error-index 取值为 0 ~ max-bindings的整数
    variable-bindings  VarBindList }       --变量 variable-binding 的类型是 VarBindList
```

Get-request 报文的 BER 编码



Get-request 报文的十六进制编码

A0 1D -- GetRequest-PDU，上下文类型，长度$1D_{16}$ = 29

02 04 05 AE 56 02 -- INTEGER类型，长度04_{16}，request-id = 05 AE 56 02

02 01 00 -- INTEGER类型，长度01_{16}，error status = 00_{16}

02 01 00 -- INTEGER类型，长度01_{16}，error index = 00_{16}

30 0F -- SEQUENCE OF类型，长度$0F_{16}$ = 15

30 0D -- SEQUENCE类型，长度$0D_{16}$ = 13

06 09 01 03 06 01 02 01 07 01 00 -- OBJECT IDENTIFIER类型，长度09_{16}，udpInDatagrams

05 00 -- NULL类型，长度00_{16}

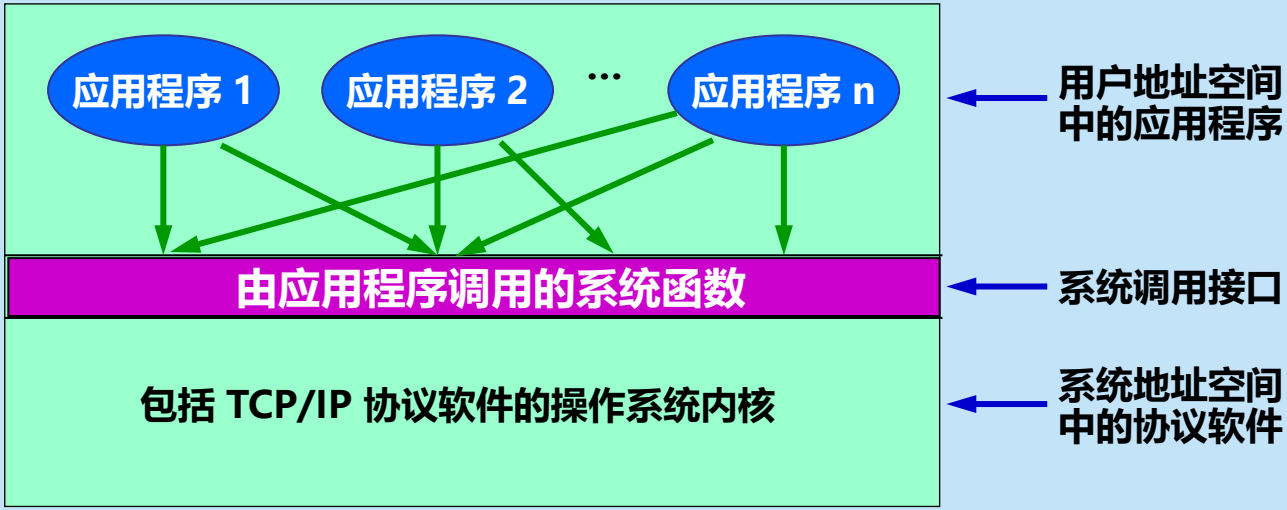
6.8 应用进程跨越网络的通信

6.8.1 系统调用和应用编程接口
6.8.2 几种常用的系统调用

6.8.1 系统调用和应用编程接口

- 大多数操作系统使用**系统调用**（system call）的机制在应用程序和操作系统之间传递控制权。
- 对程序员来说，每一个系统调用和一般程序设计中的函数调用非常相似，只是系统调用是将控制权传递给了操作系统。

多个应用进程使用系统调用的机制



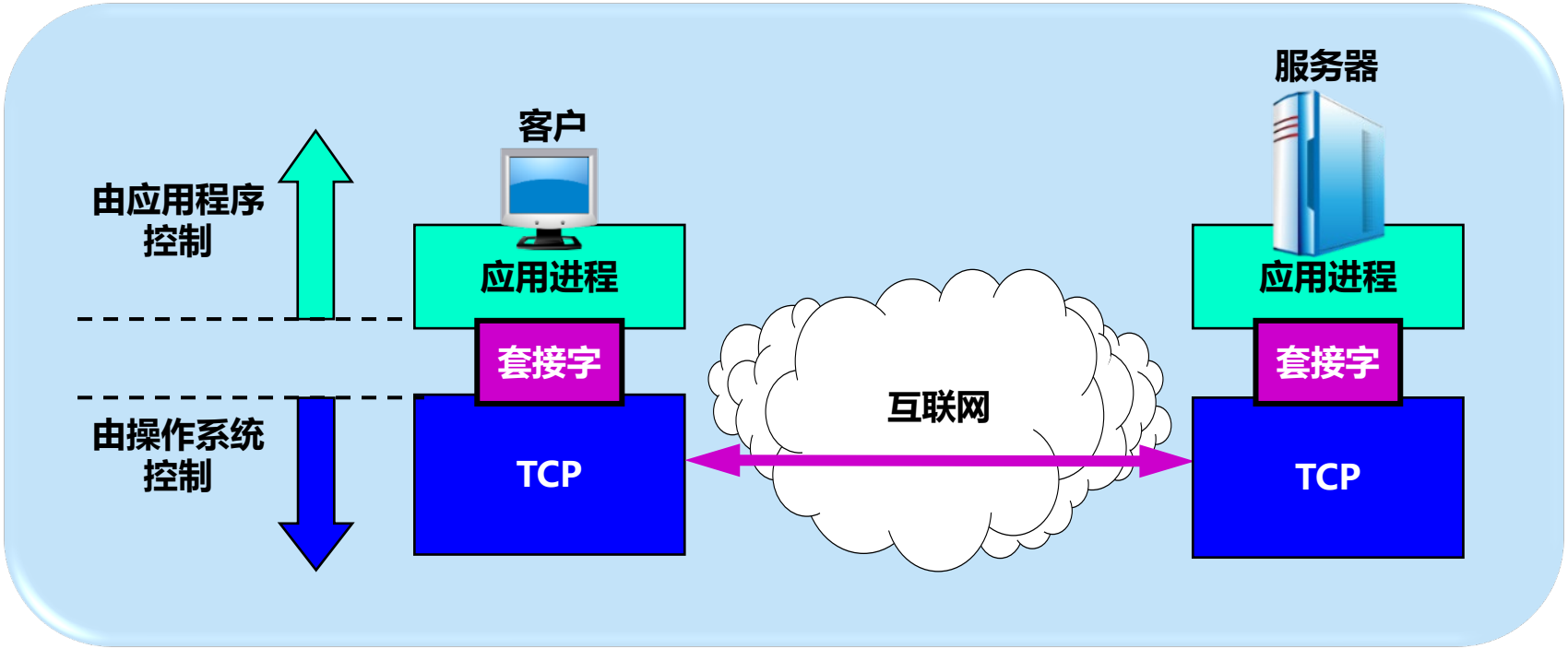
应用编程接口 API

- **系统调用接口**实际上就是应用进程的控制权和操作系统的控制权进行转换的一个接口。
- 使用系统调用之前要编写一些程序，特别是需要设置系统调用中的许多参数，因此这种系统调用接口又称为**应用编程接口 API** (Application Programming Interface)。

几种应用编程接口 API

- Berkeley UNIX 操作系统定义了一种 API，它又称为套接字接口 (socket interface)。
- 微软公司在其操作系统中采用了套接字接口 API，形成了一个稍有不同的 API，并称之为 Windows Socket。
- AT&T 为其 UNIX 系统 V 定义了一种 API，简写为 TLI (Transport Layer Interface)。

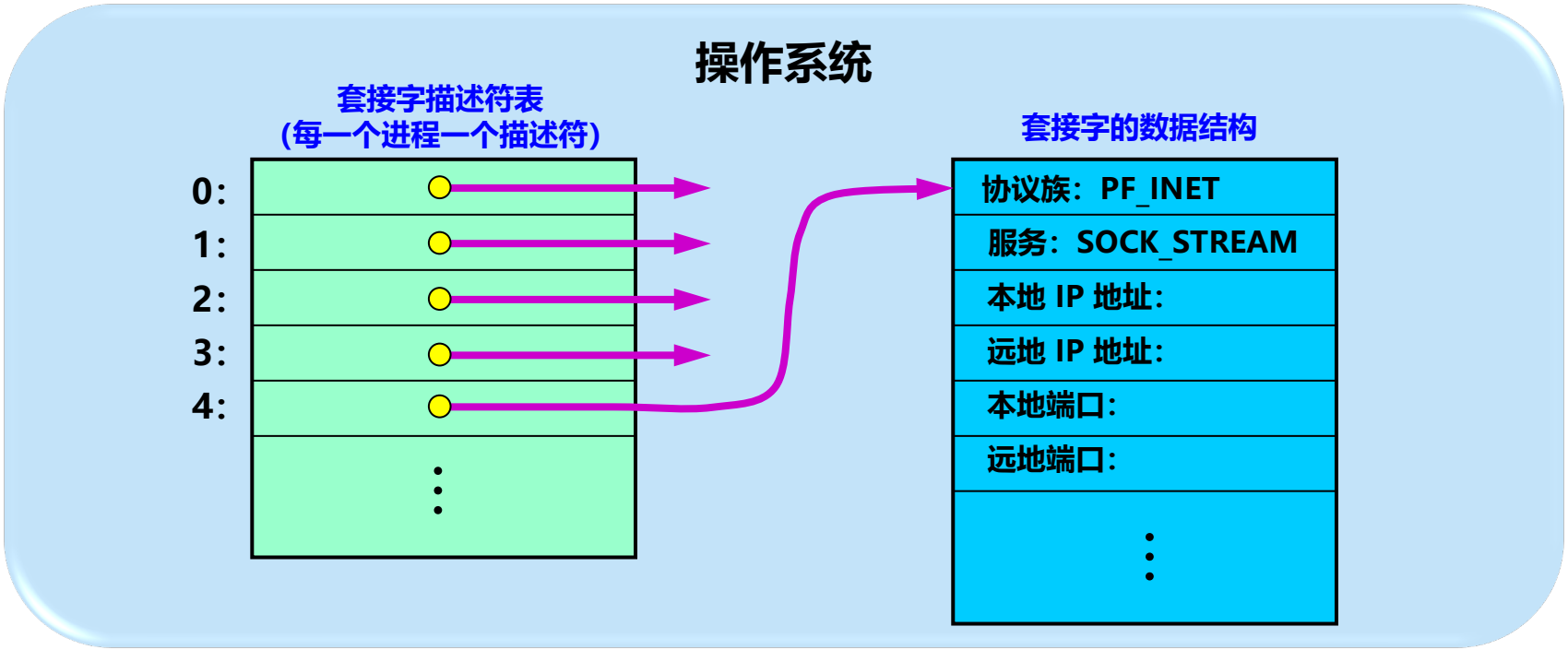
应用进程通过套接字接入到网络



套接字的作用

- 当应用进程需要使用网络进行通信时就发出系统调用，请求操作系统为其**创建套接字**，以便把网络通信所需要的系统资源分配给该应用进程。
- 操作系统为这些资源的总和用一个**套接字描述符**的号码来表示。
- 应用进程所进行的网络操作都必须使用这个套接字描述符。
- 通信完毕后，应用进程通过一个**关闭**套接字的系统调用通知操作系统回收与该套接字描述符相关的所有资源。

调用 socket 创建套接字



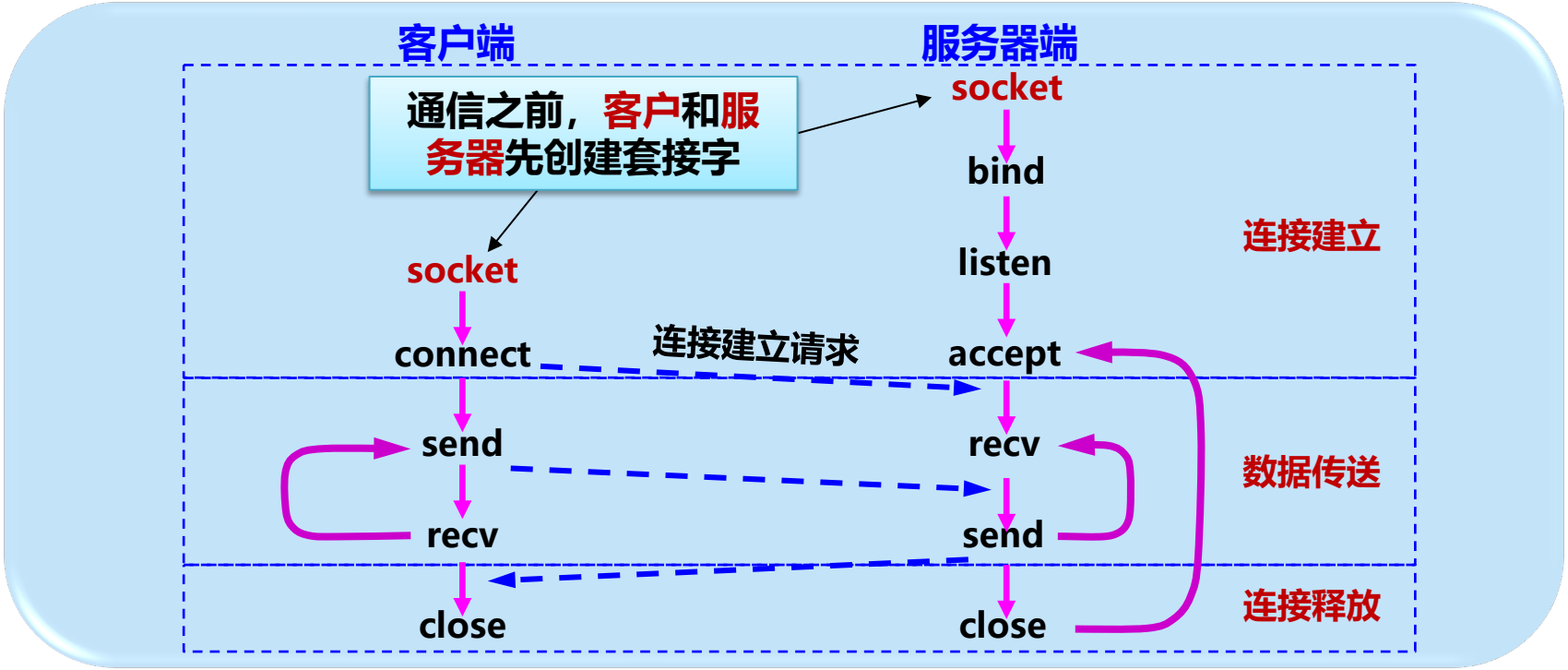
6.8.2 几种常用的系统调用

- 当应用进程需要使用网络进行通信时，就发出系统调用。
- 使用 TCP/IP 应用编程接口 API，就可以编写基于互联网的网络应用程序了。
- 调用 API 时，用户可以使用 TCP 服务，也可以使用 UDP 等其他服务。

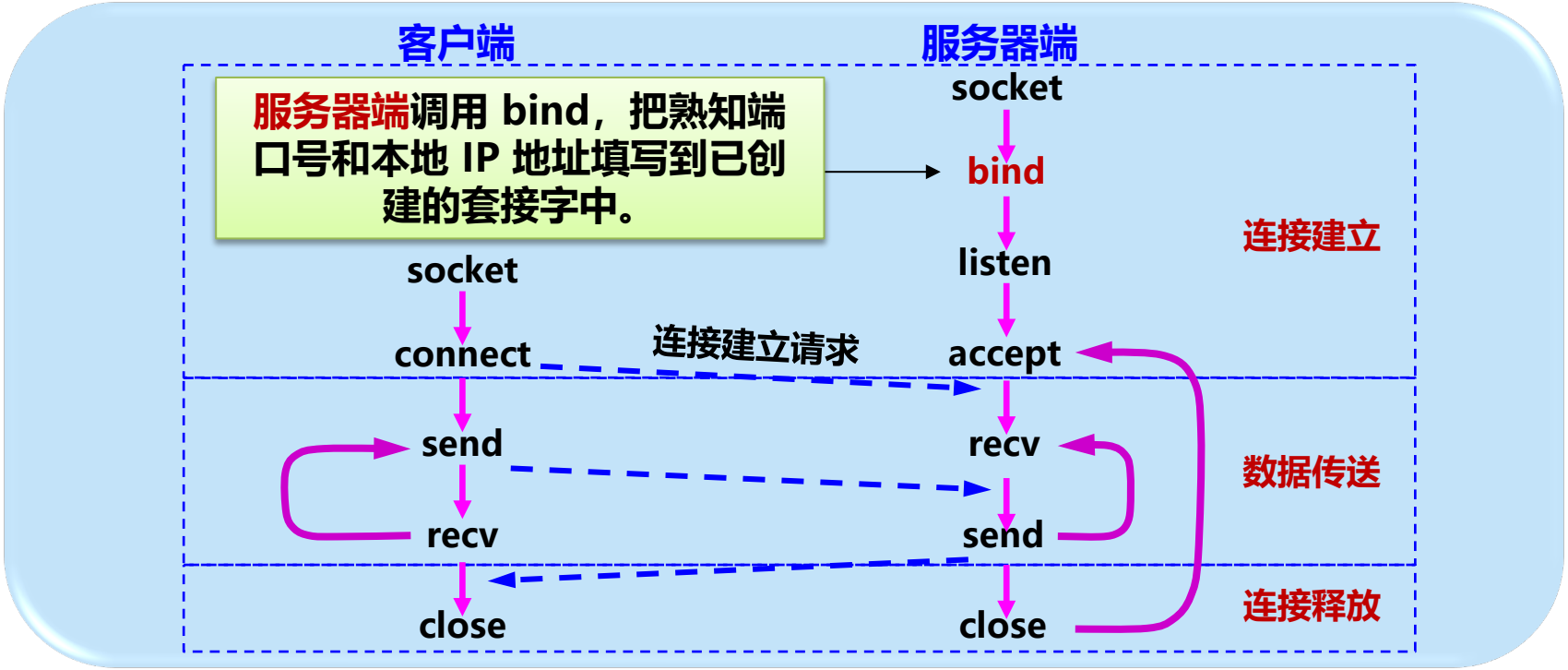
6.8.2 几种常用的系统调用

- TCP 提供面向连接的服务。
- 使用 TCP 服务需要经历 3 个阶段：
  1. 连接建立阶段
  2. 数据传送阶段
  3. 和 连接释放阶段

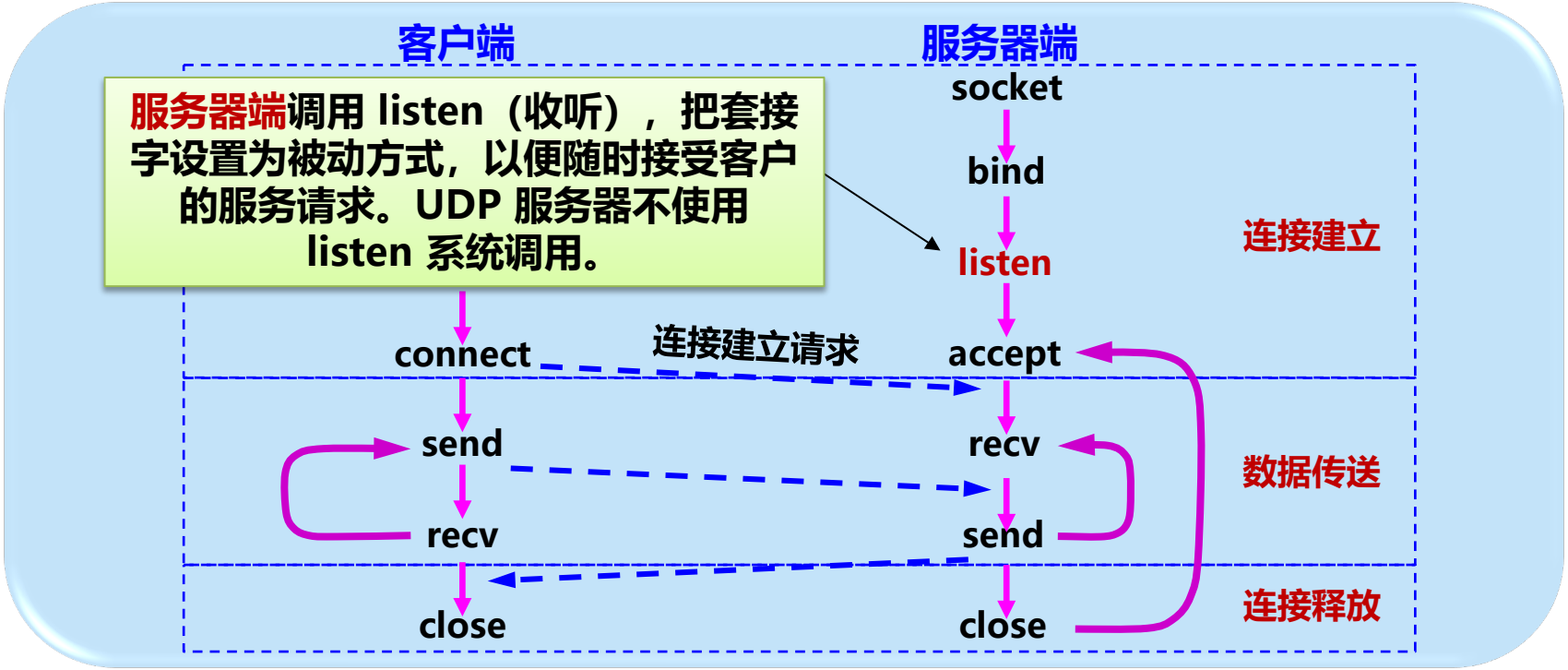
1. 连接建立阶段


客户端
服务器端
通信之前，客户和服务器先创建套接字
socket
bind
listen
accept
socket
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

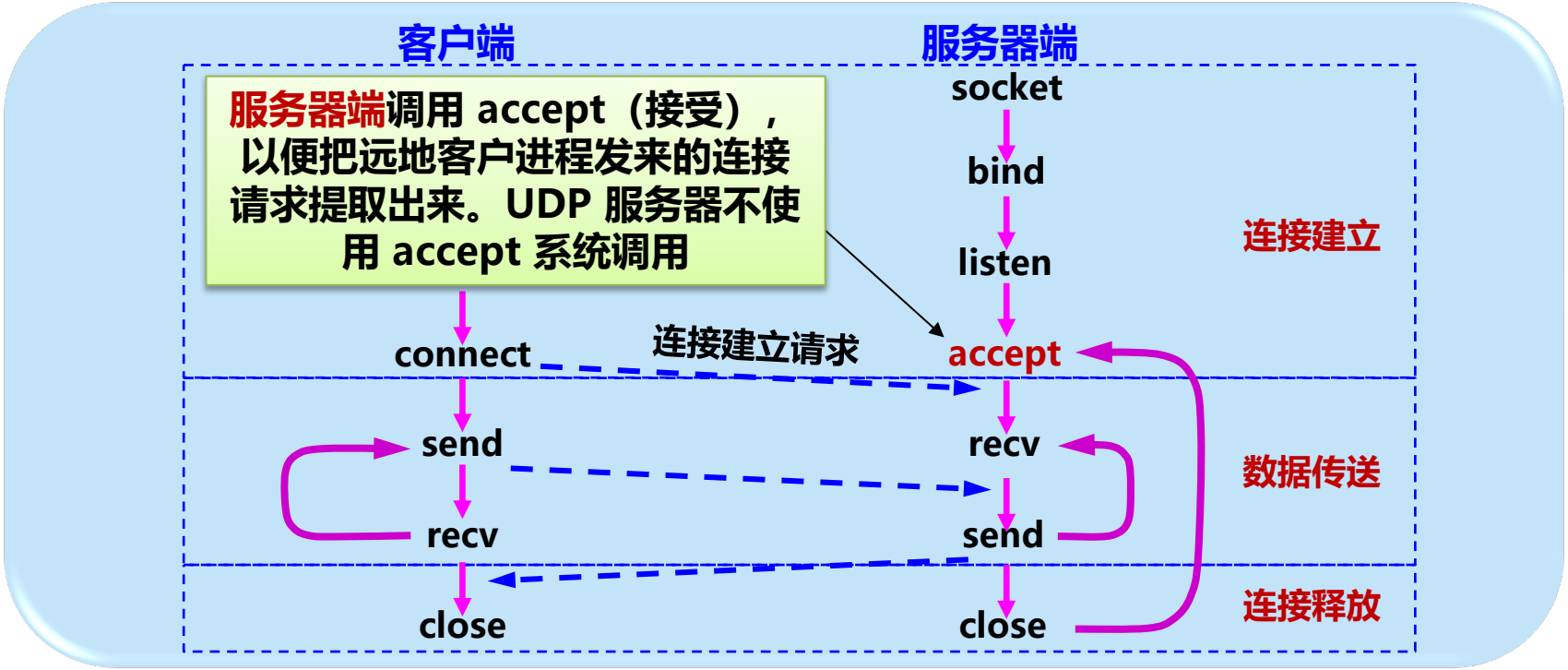
1. 连接建立阶段


客户端
服务器端
服务器端调用 bind，把熟知端口号和本地 IP 地址填写到已创建的套接字中。
socket
bind
listen
accept
socket
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

1. 连接建立阶段

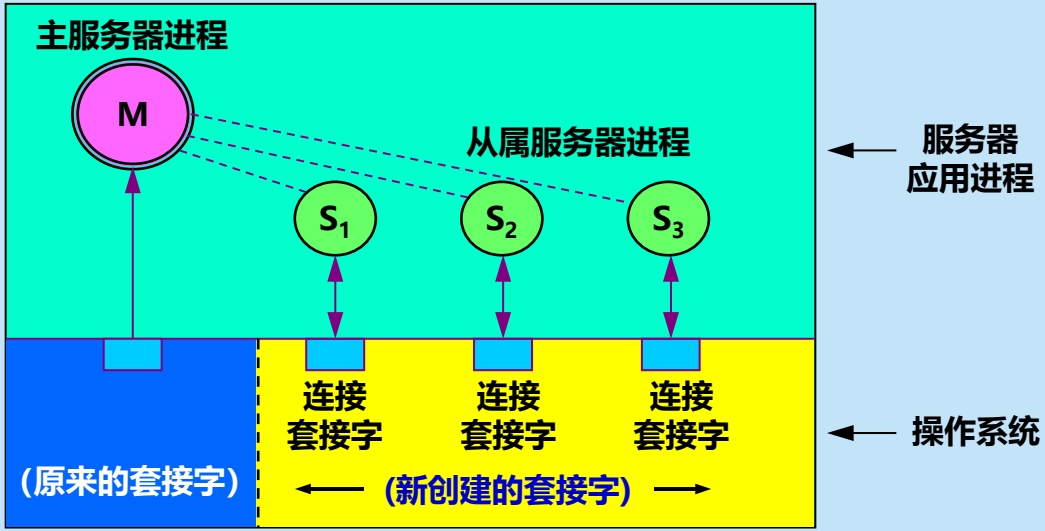

客户端
服务器端
服务器端调用 listen（收听），把套接字设置为被动方式，以便随时接受客户的服务请求。UDP 服务器不使用 listen 系统调用。
socket
bind
listen
connect
连接建立请求
accept
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

1. 连接建立阶段

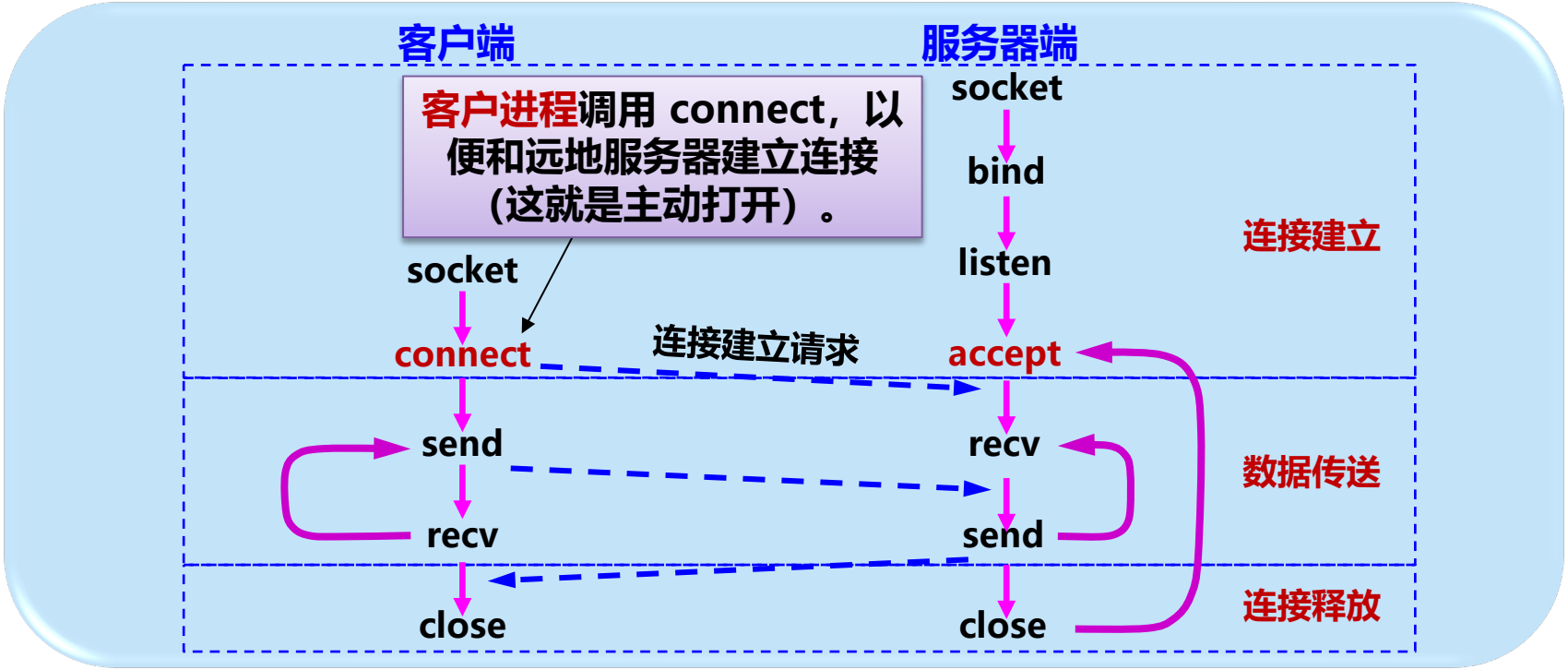

客户端
服务器端
服务器端调用 accept（接受），以便把远地客户进程发来的连接请求提取出来。UDP 服务器不使用 accept 系统调用
socket
bind
listen
accept
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

并发方式工作的服务器

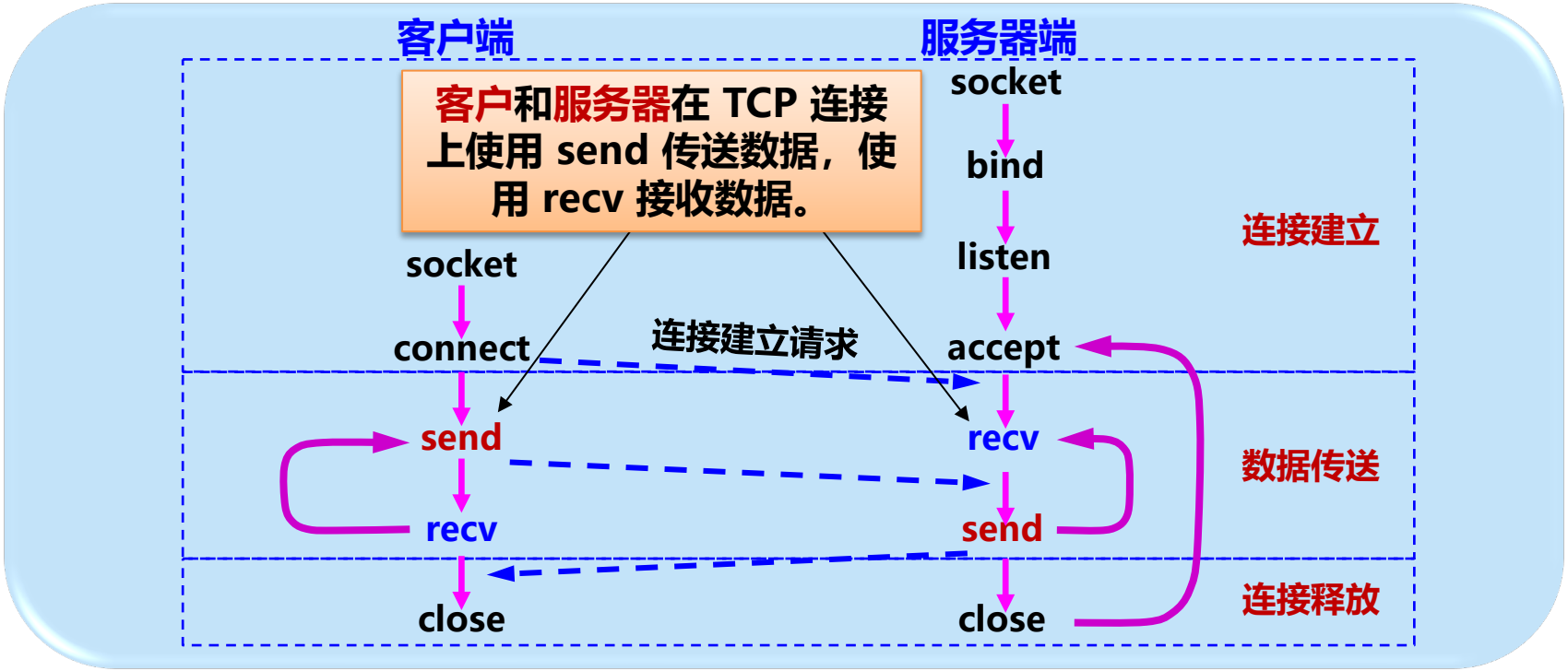
调用 accept 要完成的动作较多。这是因为一个服务器必须能够**同时**处理多个连接。这样的服务器常称为**并发方式** (concurrent) 工作的服务器。



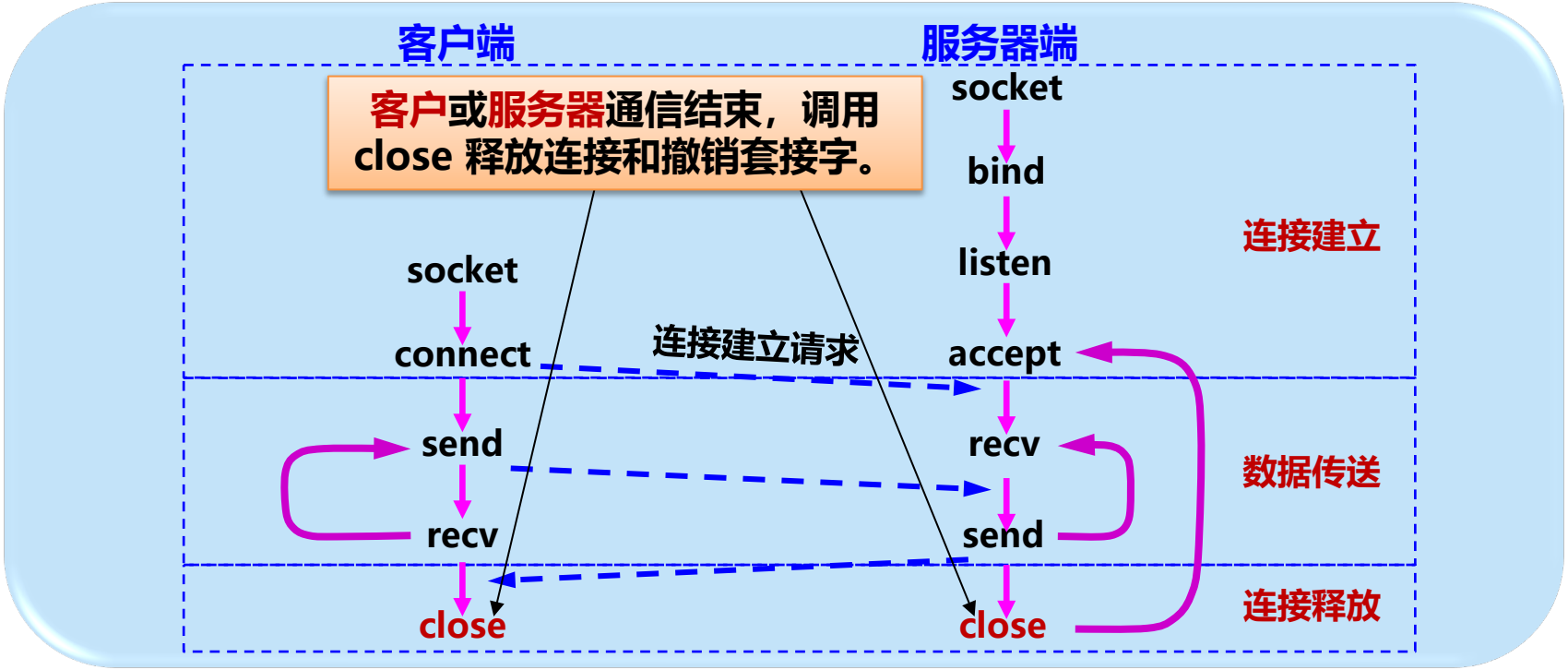
1. 连接建立阶段


客户端
服务器端
客户进程调用 connect，以便和远地服务器建立连接（这就是主动打开）。
socket
bind
listen
accept
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

2. 传送阶段


客户端
服务器端
客户和服务器在 TCP 连接上使用 send 传送数据，使用 recv 接收数据。
socket
bind
listen
accept
socket
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

3. 连接释放阶段

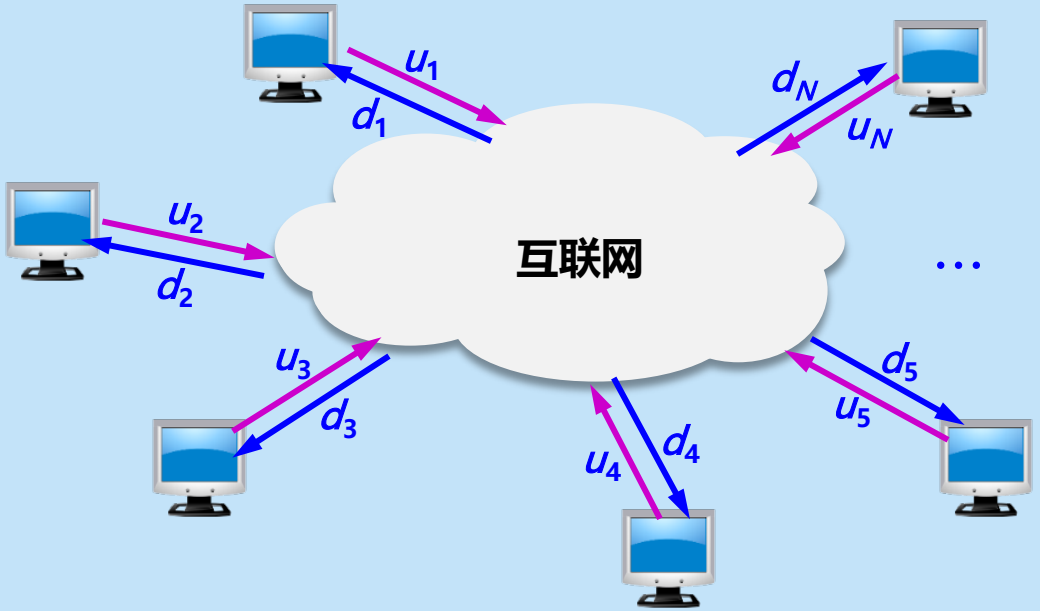

客户端
服务器端
客户或服务器通信结束，调用close 释放连接和撤销套接字。
socket
bind
listen
accept
socket
connect
连接建立请求
send
recv
recv
send
close
close
连接建立
数据传送
连接释放

6.9 P2P 应用

6.9.1 具有集中目录服务器的 P2P 工作方式
6.9.2 具有全分布式结构的 P2P 文件共享程序
6.9.3 P2P 文件分发的分析
6.9.4 在 P2P 对等方中搜索对象

P2P 工作方式概述

在 P2P 工作方式下，所有的音频/视频文件都是在普通的互联网用户之间传输。

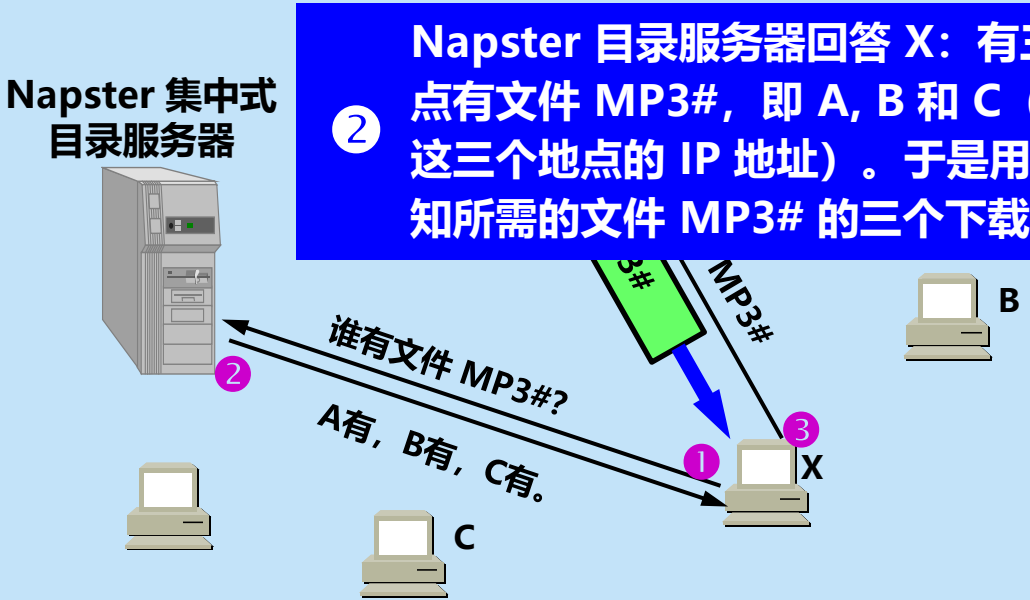


6.9.1 具有集中目录服务器的 P2P 工作方式

- Napster 最早使用 P2P 技术，提供免费下载 MP3 音乐。
- Napster 将所有音乐文件的索引信息都集中存放在 Napster 目录服务器中。
- 使用者只要查找目录服务器，就可知道应从何处下载所要的 MP3 文件。
- 用户要及时向 Napster 的目录服务器报告自己存有的音乐文件。
- Napster 的文件传输是分散的，文件的定位则是集中的。

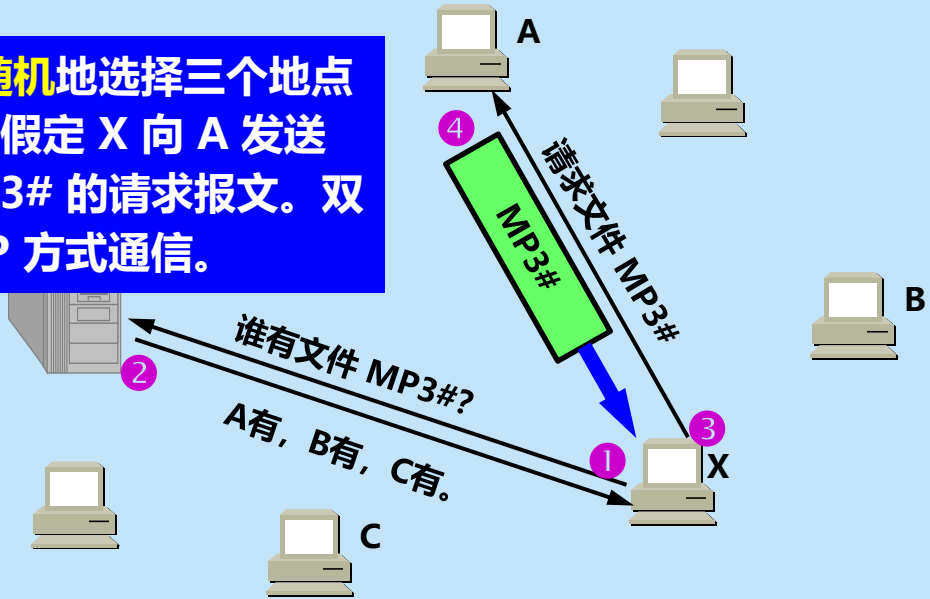
Napster 的工作过程

Napster 的工作过程



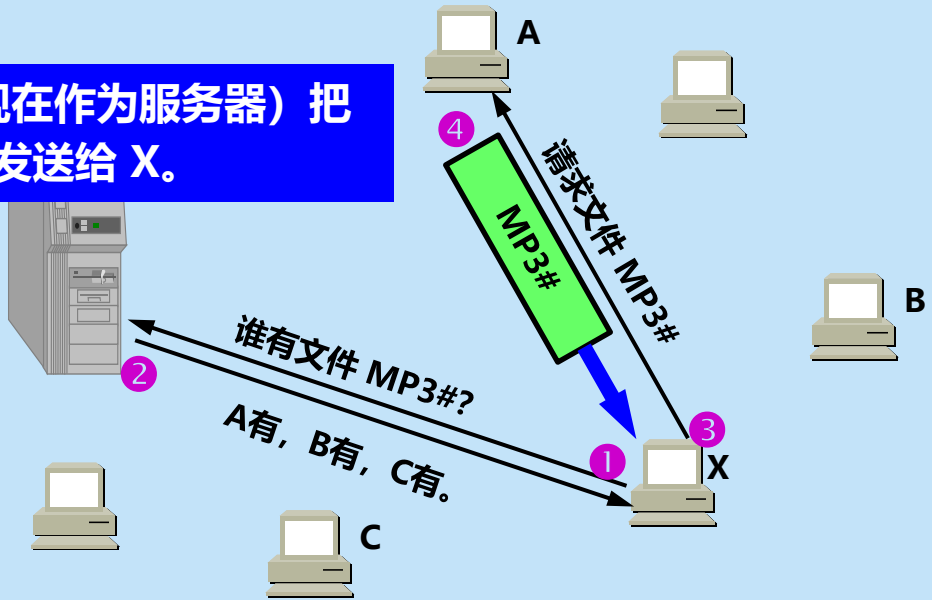
Napster 的工作过程

③ 用户 X 可以随机地选择三个地点中的任一个。假定 X 向 A 发送下载文件 MP3# 的请求报文。双方都使用 P2P 方式通信。



Napster 的工作过程

④ 对等方 A（现在作为服务器）把文件 MP3# 发送给 X。



集中式目录服务器的缺点

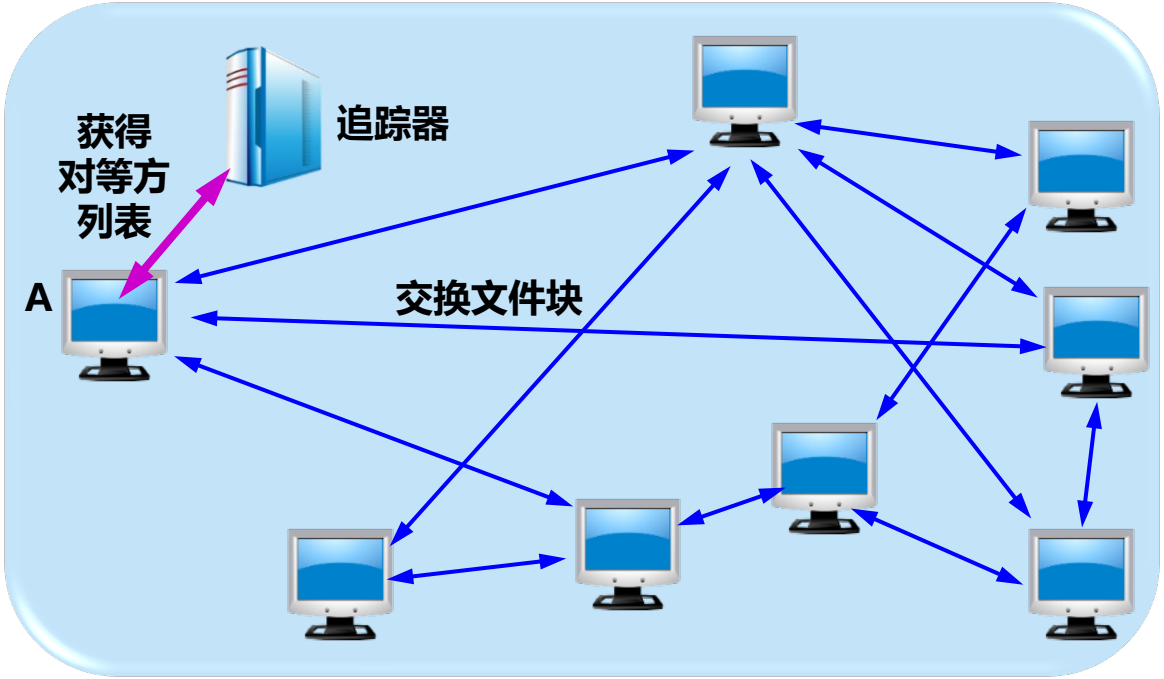
- 可靠性差。
- 会成为性能的瓶颈。

6.9.2 具有全分布式结构的 P2P 文件共享程序

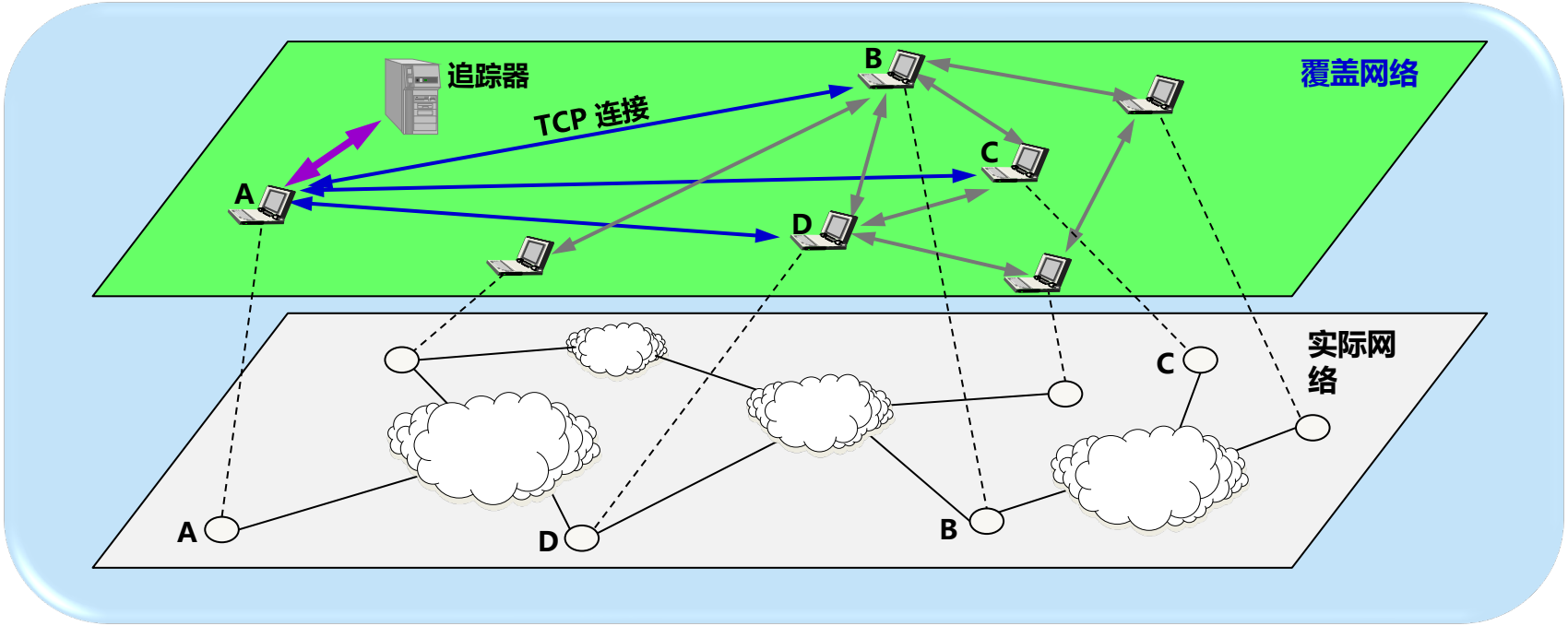
- Gnutella 是第二代 P2P 文件共享程序，采用全分布方法定位内容的 P2P 文件共享应用程序。
- Gnutella 与 Napster 最大的区别：不使用集中式的目录服务器，而是使用洪泛法在大量 Gnutella 用户之间进行查询。
- 为了不使查询的通信量过大，Gnutella 设计了一种有限范围的洪泛查询，减少了倾注到互联网的查询流量，但也影响到查询定位的准确性。
- 第三代 P2P 文件共享程序采用分散定位和分散传输技术。例如 KaZaA，电骡 eMule，比特洪流 BT (Bit Torrent) 等。

使用 P2P 的比特洪流 BT 主要特点

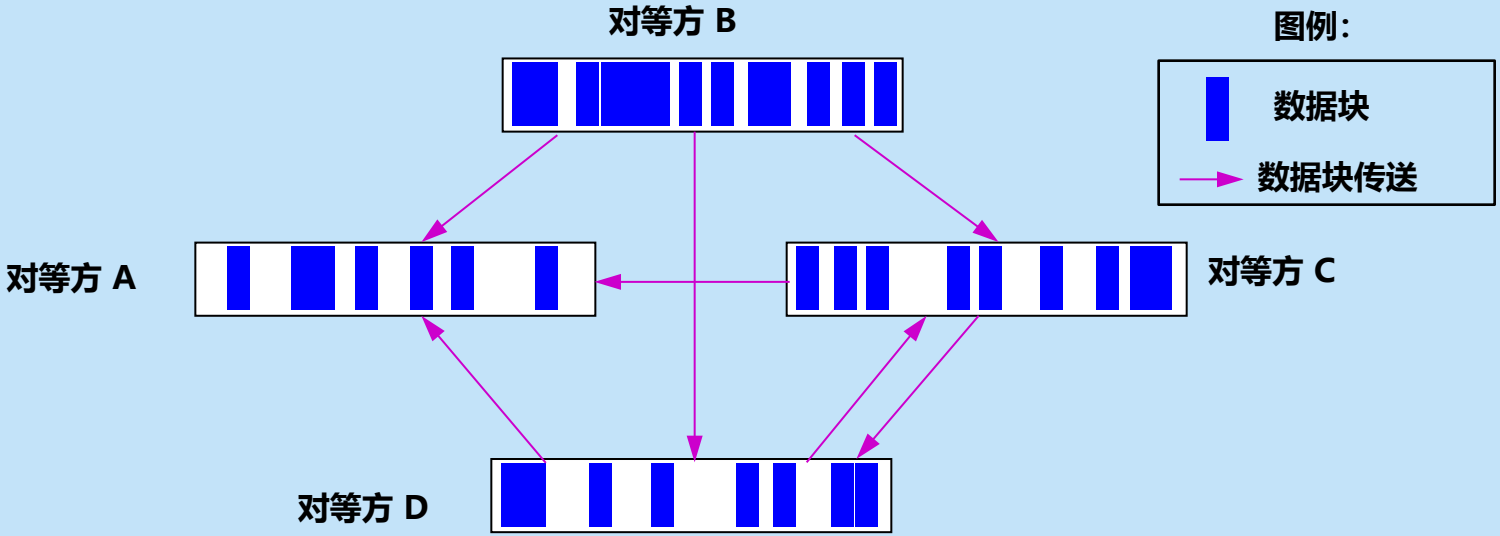
- BitTorrent 所有对等方集合称为一个洪流 (torrent)。
- 下载文件的数据单元为长度固定的文件块 (chunk)。
- 基础设施结点，叫做追踪器 (tracker)。
- A 和对等方建立了 TCP 连接。所有与 A 建立了 TCP 连接的对等方为相邻对等方(neighboring peers)。



相邻关系是逻辑的，对等方的数目是动态变化的



对等方之间互相传送文件数据块



BT 协议

- **问题：**哪些文件块是**首先**需要**向其相邻**对等方请求的?
- **方法：**A 使用**最稀有的优先** (rarest first) 的技术，首先向其相邻对等方请求对应的文件块。
- **稀有：**如果 A 所缺少的文件块在相邻对等方中的副本很少，那就是“很稀有的”。
- **问题：**在很多向 A 请求文件块的相邻对等方中，A 应当向哪些相邻对等方发送所请求的文件块?
- **方法：**凡当前以**最高数据率**向 A 传送文件块的某相邻对等方，A 就**优先**把所请求的文件块传送给该相邻对等方。

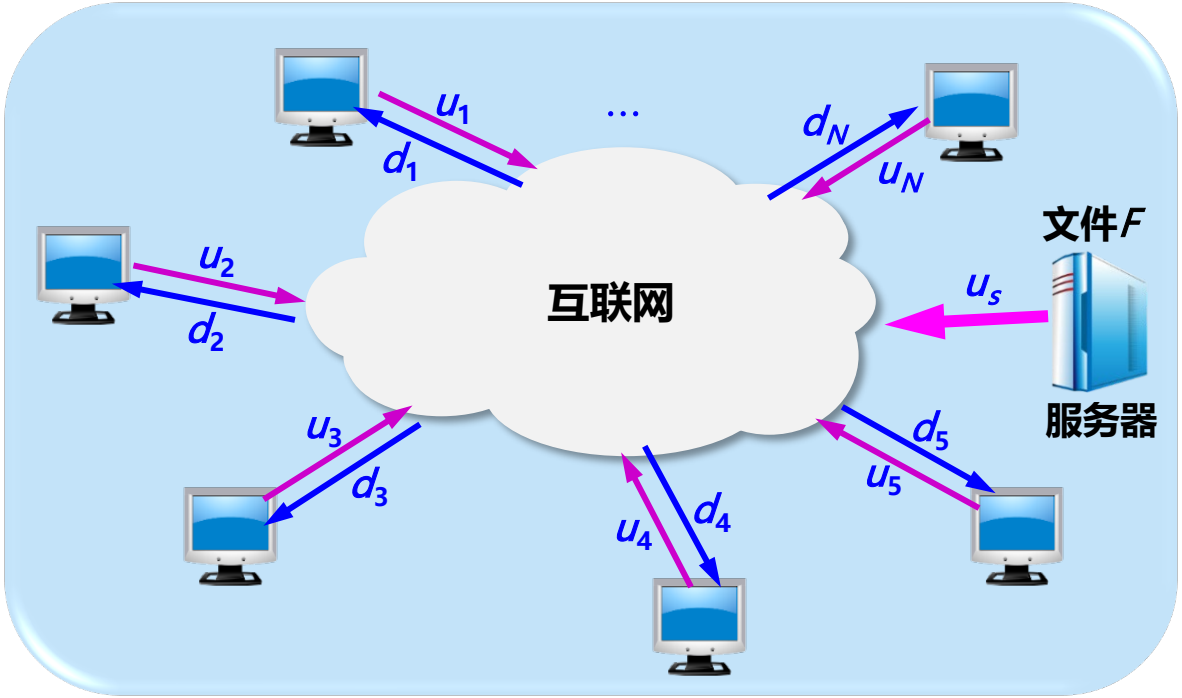
6.9.3 P2P 文件分发的分析

一些概念：

- 从互联网传送数据到主机，叫做**下载** (download)，
- 从主机向互联网传送，则称为**上传** (upload) 或**上载**。

6.9.3 P2P 文件分发的分析

- 有 N 台主机从服务器**下载**一个大文件，其长度为 F bit。
- 假定主机与互联网连接的链路的上传速率和下载速率分别为 u_i 和 d_i，单位都是 bit/s。



客户-服务器方式下分发的最短时间分析

- 从服务器端考虑，所有主机分发完毕的最短时间 T_{cs} 不可能小于 NF/u_s；
- 下载速率最慢的主机的下载速率为 d_{min}，则 T_{cs} 不可能小于 F/d_{min} 。
- 由此可得出所有主机都下载完文件 F 的最少时间是：

$$T_{cs}=\max（NF/u_s，F/d_{min}）$$

P2P 方式下分发的最短时间分析

- **初始**服务器文件分发的最少时间不可能小于 F/u_s；
- **下载**文件分发的最少时间不可能小于 F/d_{min}；
- **上载**文件分发的最少时间不可能小于 NF/u_T，其中是 u_T 是上传速率之和。
- **所有**主机都下载完文件 F 的**最少时间的下限**是：

$$T_{p2p} >= \max\left(F/u_s,\ F/d_{min},\ NF/u_T\right)$$

时间比较

- 设所有的对等方的上传速率都是 u，并且 $F/u = 1$ 小时。
- 设服务器的上传速率 $u_s = 10u$。
- 当 $N = 30$ 时，
  1. **P2P 方式：**最少时间的下限是 0.75 小时 < 1 小时（不管 N 多大）。
  2. **客户服务器方式：**最少时间是 3 小时。

6.9.4 在 P2P 对等方中搜索对象

- Napster 在一个集中式目录服务器中构建查找数据库，简单，但性能上有**瓶颈。**
- Gnutella 是一种采用全分布方法定位内容的 P2P 文件共享应用程序，它解决了集中式目录服务器所造成的瓶颈问题。但 Gnutella 是在非结构化的覆盖网络中采用查询洪泛的方法进行查找，因此查找的**效率较低。**

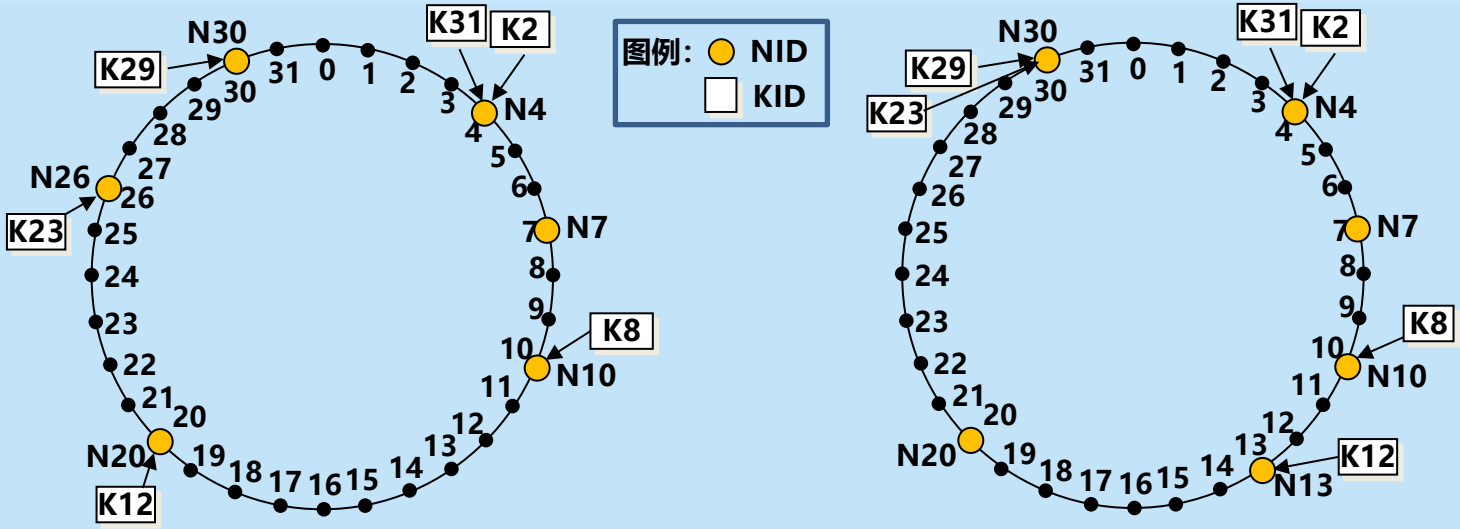
6.9.4 在 P2P 对等方中搜索对象

- 现在广泛使用的索引和查找技术叫做**分布式散列表** DHT (Distributed Hash Table)。
- DHT 也可译为**分布式哈希表**，由大量对等方共同维护。
- 广泛使用的 **Chord 算法**是美国麻省理工大学于 2001 年提出的。

基于 DHT 的 Chord 环

- 分布式散列表 DHT 利用散列函数，把资源名 K 及其存放的结点 IP 地址 N 都分别映射为资源名标识符 KID 和结点标识符 NID。
- Chord 把结点按标识符数值从小到大沿顺时针排列成一个环形覆盖网络。
- 每个资源由 Chord 环上与其标识符值最接近的下一个结点提供服务。

基于 DHT 的 Chord 环

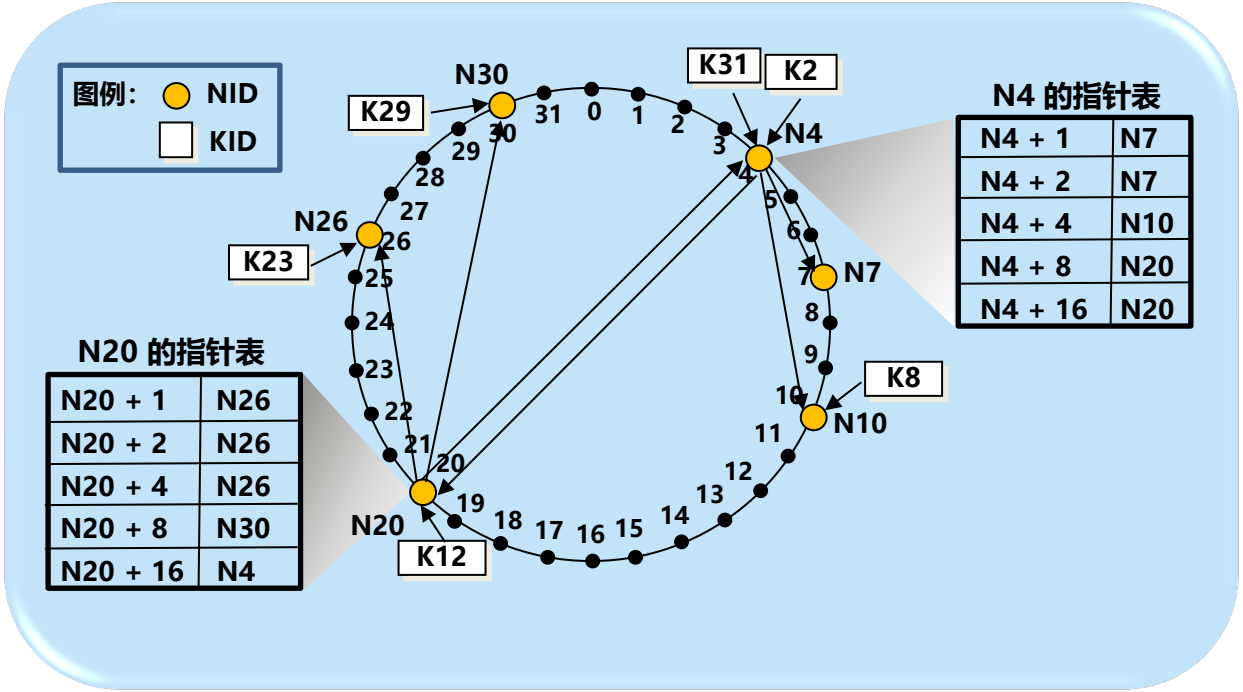


(a) KID 和 NID 映射在环上

(b) N13 加入，N26 退出

通过指针表加速 Chord 表查找

- 为了加速查找，在 Chord 环上可以增加一些指针表(finger table)，又称为路由表或查找器表。
- 对于结点 N4，其指针表的第 2 列第 i 行根据（N4 + 2^{i-1}）计算得出其后继结点。



N4 的指针表

N4 + 1	N7
N4 + 2	N7
N4 + 4	N10
N4 + 8	N20
N4 + 16	N20

N20 的指针表

N20 + 1	N26
N20 + 2	N26
N20 + 4	N26
N20 + 8	N30
N20 + 16	N4