第17章

内容协商与转码

一个 URL 常常需要代表若干不同的资源。例如那种需要以多种语言提供其内容的网站站点。如果某个站点（比如 Joe 的五金商店这样的站点）有说法语的和说英语的两种用户，它可能想用这两种语言提供网站站点信息。但在这种情况下，当用户请求 http://www.joes-hardware.com 时，服务器应当发送哪种版本呢？法文版还是英文版？

理想情况下，服务器应当向英语用户发送英文版，向法语用户发送法文版——用户只要访问 Joe 的五金商店的主页就可以得到相应语言的内容。幸运的是，HTTP 提供了内容协商方法，允许客户端和服务器作这样的决定。通过这些方法，单一的 URL 就可以代表不同的资源（比如，同一个网站页面的法语版和英语版）。这些不同的版本称为变体。

对于特定的 URL 来说，服务器还可以根据其他原则来决定发送什么内容给客户端最合适。在有些场合下，服务器甚至可以自动生成定制的页面。比如，服务器可以为手持设备把 HTML 页面转换成 WML 页面。这类动态内容变换被称为转码。这些变换动作是 HTTP 客户端和服务器之间进行内容协商的结果。

本章，我们将讨论内容协商和网站应用程序应该如何担负内容协商的责任。

17.1 内容协商技术

共有 3 种不同的方法可以决定服务器上哪个页面最适合客户端：让客户端来选择、服务器自动判定，或让中间代理来选。这 3 种技术分别称为客户端驱动的协商、服务器驱动的协商以及透明协商（参见表 17-1）。本章，我们将研究每种技术的机制及其优缺点。

395

表17-1 内容协商技术概要

技　术	工作原理	优　点	缺　点
客户端驱动	客户端发起请求，服务器发送可选项的列表，客户端选择	在服务器端的实现最容易。客户端可以选择最合适的内容	增加了时延：为了获得正确的内容，至少要发送两次请求
服务器驱动	服务器检查客户端的请求首部集并决定提供哪个版本的页面	比客户端驱动的协商方式要快。HTTP 提供了 q 值机制，允许服务器近似匹配，还提供了 Vary 首部供服务器告知下游的设备如何对请求估值	如果结论不是很明确（比如首部集不匹配），服务器要做猜测
透明	某个中间设备（通常是缓存代理）代表客户端进行请求协商	免除了 Web 服务器的协商开销。比客户端驱动的协商要快	关于如何进行透明协商，还没有正式的规范

17.2 客户端驱动的协商

对于服务器来说，收到客户端请求时只是发回响应，在其中列出可用的页面，让客户端决定要看哪个，这是最容易的事情。很显然，这是服务器最容易实现的方式，而且客户端很可能选择到最佳的版本（只要列表中有让客户端选择的足够信息）。不利之处是每个页面都需要两次请求：第一次获取列表，第二次获取选择的副本。这种技术速度很慢且过程枯燥乏味，让用户厌烦。

从实现原理上来说，服务器实际上有两种方法为客户端提供选项：一是发送回一个HTML 文档，里面有到该页面的各种版本的链接和每个版本的描述信息；另一种方法是发送回 HTTP/1.1 响应时，使用 300 Multiple Choices 响应代码。客户端浏览器收到这种响应时，在前一种情况下，会显示一个带有链接的页面；在后一种情况下，可能会弹出对话窗口，让用户做选择。不管怎么样，决定是由客户端的浏览器用户作出的。

除了增加时延并且对每个页面都要进行繁琐的多次请求之外，这种方法还有一个缺点：它需要多个 URL：公共页面要一个，其他每种特殊页面也都要一个。因此，比如说原始的请求地址是 www.joes-hardware.com，Joe 的服务器可能会回复某个页面，该页面里面有到 www.joes-hardware.com/english 和 www.joes-hardware.com/french 的链接。如果客户端想加书签的话，是要加在原始的公共页面上呢，还是加在选中的页面上呢？如果用户想把这个网站推荐给他的朋友，是告知 www.joes-hardware.com 这个地址好呢，还是只告诉他们讲英语的朋友 www.joes-hardware.com/english 这个地址？ 396

17.3 服务器驱动的协商

在前一节中，我们了解了客户端驱动的协商存在的若干缺点。大部分缺点都涉及客户端和服务器之间通信量的增长，这些通信量用来决定什么页面才是对请求的最佳响应。减少额外通信量的一种方法是让服务器来决定发送哪个页面回去，但为了做到这一点，客户端必须发送有关客户偏好的足够信息，以便服务器能够作出准确的决策。服务器通过客户端请求的首部集来获得这方面的信息。

有以下两种机制可供 HTTP 服务器评估发送什么响应给客户端比较合适。

- 检查内容协商首部集。服务器察看客户端发送的 Accept 首部集，设法用相应的响应首部与之匹配。
- 根据其他（非内容协商）首部进行变通。例如，服务器可以根据客户端发送的 User-Agent 首部来发送响应。

后面的小节将详细介绍这两种机制。

17.3.1 内容协商首部集

客户端可以用表 17-2 中列出的 HTTP 首部集发送用户的偏好信息。

表17-2 Accept首部集

首　部	描　述
Accept	告知服务器发送何种媒体类型
Accept-Language	告知服务器发送何种语言
Accept-Charset	告知服务器发送何种字符集
Accept-Encoding	告知服务器采用何种编码

注意，这些首部与第 15 章讨论的那些实体首部非常类似。不过，这两种首部的用途截然不同。正如第 15 章中所述，实体首部集像运输标签，它们描述了把报文从服务器传输给客户端的过程中必须的各种报文主体属性。而内容协商首部集是由客户端发送给服务器用于交换偏好信息的，以便服务器可以从文档的不同版本中选择出最符合客户端偏好的那个来提供服务。

397 服务器用表 17-3 中列出的实体首部集来匹配客户端的 Accept 首部集。

表17-3 Accept首部集和匹配的文档首部集

Accept首部	实体首部
Accept	Content-Type
Accept-Language	Content-Language
Accept-Charset	Content-Type
Accept-Encoding	Content-Encoding

注意，由于 HTTP 是无状态的协议（表示服务器不会在不同的请求之间追踪客户端的偏好），所以客户端必须在每个请求中都发送其偏好信息。

如果两个客户端都发送了 Accept-Language 首部，描述它们感兴趣的语言信息，服务器就能够决定发送 www.joes-hardware.com 的何种版本给哪个客户端了。让服务器自动选择发送回去的文档，减少了往返通信的时延，这种时延是客户端驱动模型中无法避免的。

然而，假设某个客户端偏好西班牙文，那服务器应当回送哪个版本的页面呢？英语还是法语？服务器只有两种选择：猜测或回退到客户端驱动模型，问客户端选择哪个。假如这个西班牙人碰巧懂一点英语，他可能会选择英文页面，这不是最理想的，

但它能解决问题。在这种情况下，这个西班牙人需要有办法传达更多与其偏好有关的信息，也就是他的确对英语略知一二，在没有西班牙语的时候，英语也行。

幸运的是，HTTP 提供了一种机制，可以让与这个西班牙人情况类似的客户端更详细地描述其偏好。这种机制就是*质量值*（简称 q 值）。

17.3.2 内容协商首部中的质量值

HTTP 协议中定义了质量值，允许客户端为每种偏好类别列出多种选项，并为每种偏好选项关联一个优先次序。例如，客户端可以发送下列形式的 Accept-Language 首部：

```
Accept-Language: en;q=0.5, fr;q=0.0, nl;q=1.0, tr;q=0.0
```

其中 q 值的范围从 0.0 ~ 1.0（0.0 是优先级最低的，而 1.0 是优先级最高的）。上面列出的那个首部，说明该客户端最愿意接收荷兰语（缩写为 nl）文档，但英语（缩写为 en）文档也行；无论如何，这个客户端都不愿意收到法语（缩写为 fr）或土耳其语（缩写为 tr）的版本。注意，偏好的排列顺序并不重要，只有与偏好相关的 q 值才是重要的。

服务器偶尔也会碰到找不到文档可以匹配客户端的任何偏好的情况。对于这种情况，服务器可以修改文档，也就是对文档进行转码，以匹配客户端的偏好。我们将在本章后面讨论这种机制。

398

17.3.3 随其他首部集而变化

服务器也可以根据其他客户端请求首部集来匹配响应，比如 User-Agent 首部。例如，服务器知道老版本的浏览器不支持 JavaScript 语言，这样就可以向其发送不含有 JavaScript 的页面版本。

在这种情况下，没有 q 值机制可供查找“最近似”的匹配。服务器或者去找完全匹配，或者简单地有什么就给什么，这取决于服务器的实现。

由于缓存需要尽力提供所缓存文档中正确的“最佳”版本，HTTP 协议定义了服务器在响应中发送的 Vary 首部。这个首部告知缓存（还有客户端和所有下游的代理）服务器根据哪些首部来决定发送响应的最佳版本。本章后面会更详细地讨论 Vary 首部。

17.3.4 Apache中的内容协商

这里概括了著名的 Web 服务器 Apache 是如何支持内容协商的。网站的内容提供者，

比如说 Joe，要负责为 Joe 的索引页面提供不同的版本。Joe 还必须把这些索引页面文件放在和站点相关的 Apache 服务器的适当目录下。用以下两种方式可以启用内容协商。

- 在网站目录中，为网站中每个有变体的 URI 创建一个 type-map（类型映射）文件。这个 type-map 文件列出了每个变体和其相关的内容协商首部集。
- 启用 MultiViews 指令，这样会使 Apache 自动为目录创建 type-map 文件。

1. 使用type-map文件

Apache 服务器需要知道 type-map 文件的命名规则。可以在服务器的配置文件中设置 handler 来说明 type-map 文件的后缀名。例如：

```
AddHandler type-map .var
```

这行就说明了后缀是 .var 的文件就是 type-map 文件。

399 这里给出一个 type-map 文件示例：

```
URI: joes-hardware.html

URI: joes-hardware.en.html
Content-type: text/html
Content-language: en
URI: joes-hardware.fr.de.html
Content-type: text/html;charset=iso-8859-2
Content-language: fr, de
```

根据这个 type-map 文件，Apache 服务器就知道要发送 joes-hardware.en.html 给请求英语版的客户端，发送 joes-hardware.fr.de.html 给请求法语版的客户端。Apache 服务器也支持质量值，具体信息请参阅它的文档。

2. 使用MultiView

为了使用 MultiView，必须在网站目录下的 access.conf 文件中的适当小节（*<Directory>*、*<Location>*，或 *<Files>*）使用 OPTION 指令来启用它。

如果启用了 MultiView，而浏览器又请求了名为 joes-hardware 的资源，服务器就会查找所有名字中含有 joes-hardware 的文件，并为它们创建 type-map 文件。服务器会根据名字猜测其对应的内容协商首部集。例如，法语版的 joes-hardware 应当含有 *.fr*。

17.3.5 服务器端扩展

另一种在服务器端实现内容协商的方法是使用服务器端扩展，比如微软的动态服务

器页面（Microsoft's Active Server Pages，ASP）。参见第 8 章中关于服务器端扩展的综述。

17.4 透明协商

透明协商机制试图从服务器上去除服务器驱动协商所需的负载，并用中间代理来代表客户端以使与客户端的报文交换最小化。假定代理了解客户端的预期，这样就可以代表客户端与服务器协商（在客户端请求内容的时候，代理已经收到了客户端的预期）。为了支持透明内容协商，服务器必须有能力告知代理，服务器需要检查哪些请求首部，以便对客户端的请求进行最佳匹配。HTTP/1.1 规范中没有定义任何透明协商机制，但定义了 Vary 首部。服务器在响应中发送了 Vary 首部，以告知中间节点需要使用哪些请求首部进行内容协商。

代理缓存可以为通过单个 URL 访问的文档保存不同的副本。如果服务器把它们的决策过程传给缓存，这些代理就能代表服务器与客户端进行协商。缓存同时也是进行内容转码的好地方，因为部署在缓存里的通用转码器能对任意服务器，而不仅仅是一台服务器传来的内容进行转码。图 17-3 中展示了缓存对内容进行转码的情况，本章后面会更详细地探讨。 400

17.4.1 进行缓存与备用候选

对内容进行缓存的时候是假设内容以后还可以重用。然而，为了确保对客户端请求回送的是正确的已缓存响应，缓存必须应用服务器在回送响应时所用到的大部分决策逻辑。

前一节描述了客户端发送的 Accept 首部集，以及为了给每条请求选择最佳的响应，服务器使用的与这些首部集匹配的相应实体首部集。缓存也必须使用相同的首部集来决定回送哪个已缓存的响应。

图 17-1 展示了涉及缓存的正确及错误的操作序列。缓存把第一个请求转发给服务器，并存储其响应。对于第二个请求，缓存根据 URL 查找到了匹配的文档。但是，这份文档是法语版的，而请求者想要的是西班牙语版的。如果缓存只是把文档的法语版本发给请求者的话，它就犯了错误。

因此，缓存也应该把第二条请求转发给服务器，并保存该 URL 的响应与“备用候选”响应。缓存现在就保存了同一个 URL 的两份不同的文档，与服务器上一样。这 401
些不同的版本称为*变体*（variant）或*备用候选*（alternate）。内容协商可看成是为客户端请求选择最合适变体的过程。

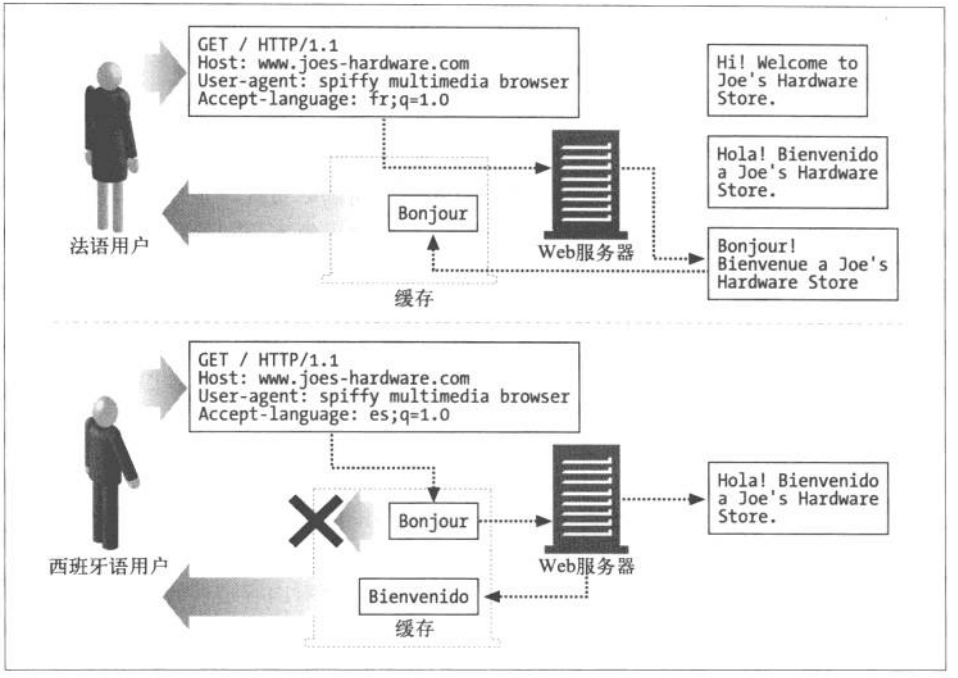

图 17-1　缓存根据内容协商首部发送给客户端正确的响应

17.4.2　Vary首部

这里是浏览器和服务器发送的一些典型的请求及响应首部：

```
GET http://www.joes-hardware.com/ HTTP/1.0
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.73 [en] (WinNT; U)
Host: www.joes-hardware.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/
    png, */*
Accept-Encoding: gzip
Accept-Language: en, pdf
Accept-Charset: iso-8859-1, *, utf-8

HTTP/1.1 200 OK
Date: Sun, 10 Dec 2000 22:13:40 GMT
Server: Apache/1.3.12 OpenSSL/0.9.5a (Unix) FrontPage/4.0.4.3
Last-Modified: Fri, 05 May 2000 04:42:52 GMT
Etag: "1b7ddf-48-3912514c"
Accept-Ranges: Bytes
Content-Length: 72
Connection: close
Content-Type: text/html
```

然而，如果服务器的决策不是依据 Accept 首部集，而是比如 User-Agent 首部的话，情况会如何？这不像听起来这么极端。例如，服务器可能知道老版本的浏览器不支持 JavaScript 语言，因此可能会回送不包含 JavaScript 的页面版本。如果服务器是根据其他首部来决定发送哪个页面的话，缓存必须知道这些首部是什么，这样才能在选择回送的页面时做出同样的逻辑判断。

HTTP 的 Vary 响应首部中列出了所有客户端请求首部，服务器可用这些首部来选择文档或产生定制的内容（在常规的内容协商首部集之外的内容）。例如，若所提供的文档取决于 User-Agent 首部，Vary 首部就必须包含 User-Agent。

当新的请求到达时，缓存会根据内容协商首部集来寻找最佳匹配。但在把文档提供给客户端之前，它必须检查服务器有没有在已缓存响应中发送 Vary 首部。如果有 Vary 首部，那么新请求中那些首部的值必须与旧的已缓存请求里相应的首部相同。因为服务器可能会根据客户端请求的首部来改变响应，为了实现透明协商，缓存必须为每个已缓存变体保存客户端请求首部和相应的服务器响应首部，参见图 17-2。 402

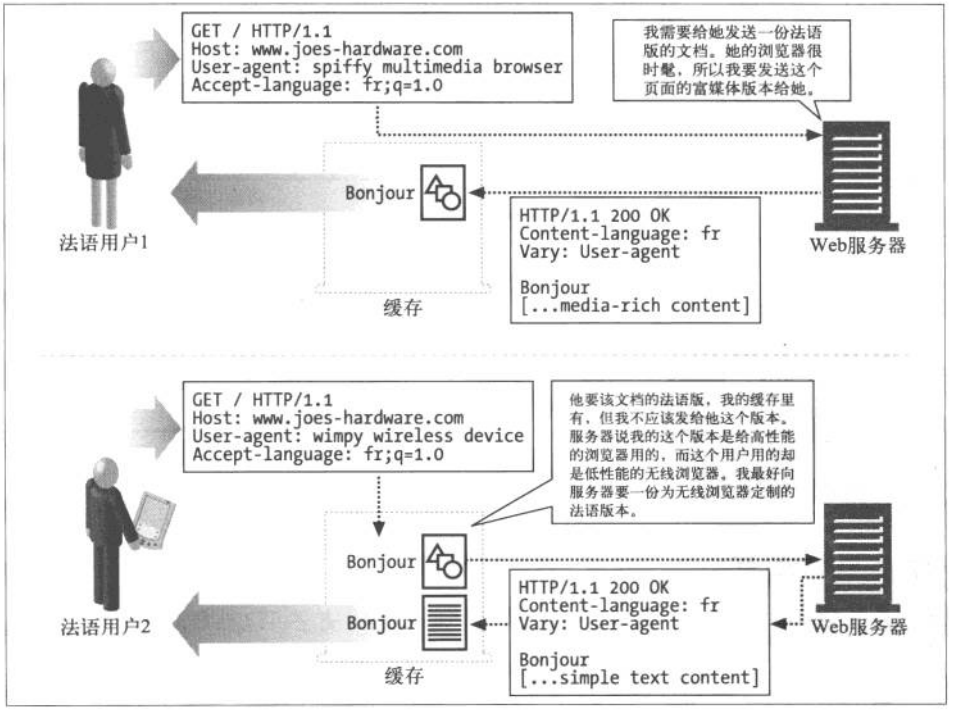

图 17-2 如果服务器根据特定的请求首部集来选择变体，缓存必须在发送回缓存的响应之前，检查常规的内容协商首部集和这些请求首部

如果某服务器的 Vary 首部看起来像下面这样，大量不同的 User-Agent 和 Cookie 值将会产生非常多的变体：

```
Vary: User-Agent, Cookie
```

缓存必须为每个变体保存其相应的文档版本。当缓存执行查找时，首先会对内容协商首部集进行内容匹配，然后比较请求的变体与缓存的变体。如果无法匹配，缓存就从原始服务器获取文档。

17.5 转码

我们已经讨论了一个机制，该机制可以让客户端和服务器从某个 URL 的一系列文档
403 中挑选出最适合客户端的文档。实现这些机制的前提是，存在一些满足客户端需求的文档——不管是完全满足还是在一定程度上满足。

然而，如果服务器没有能满足客户端需求的文档会怎么样呢？服务器可以给出一个错误响应。但理论上，服务器可以把现存的文档转换成某种客户端可用的文档。这种选项称为*转码*。

表 17-4 列出了一些假设的转码。

表17-4 假设的转码

转换之前	转换之后
HTML 文档	WML 文档
高分辨率图像	低分辨率图像
彩色图像	黑白图像
有多个框架的复杂页面	没有很多框架或图像的简单文本页面
有 Java 小应用程序的 HTML 页面	没有 Java 小应用程序的 HTML 页面
有广告的页面	去除广告的页面

有 3 种类别的转码：格式转换、信息综合以及内容注入。

17.5.1 格式转换

*格式转换*是指将数据从一种格式转换成另一种格式，使之可以被客户端查看。通过 HTML 到 WML 的转换，无线设备就可以访问通常供桌面客户端查看的文档了。通过慢速连接访问 Web 页面的客户端并不需要接收高分辨率图像，如果通过格式转换降低图像分辨率和颜色来减小图像文件大小的话，这类客户端就能更容易地查看图像比较丰富的页面了。

格式转换可以由表 17-2 中列出的内容协商首部集来驱动，但也能由 User-Agent 首部来驱动。注意，内容转换或转码与内容编码或传输编码是不同的，后两者一般用于更高效或安全地传输内容，而前两者则可使访问设备能够查看内容。

17.5.2 信息综合

从文档中提取关键的信息片段称为信息综合（information synthesis），这是一种有用的转码操作。这种操作的例子包括根据小节标题生成文档的大纲，或者从页面中删除广告和商标。

404

根据内容中的关键字对页面分类是更精细的技术，有助于总结文档的精髓。这种技术常用于 Web 页面分类系统中，比如门户网站的 Web 页面目录。

17.5.3 内容注入

前面描述的两类转码通常会减少 Web 文档的内容，但还有另一类转换会增加文档的内容，即内容注入转码。内容注入转码的例子有自动广告生成器和用户追踪系统。

设想一下，一个能往途经的每个 HTML 页面中自动添加广告的广告植入转码器是多么的诱人（当然也很烦人）。这类转码操作只能动态进行——它必须即时添加与当前的特定用户有关，或针对特定用户的广告。也可以构建用户追踪系统，在页面中动态增加内容，用于收集用户查看页面和客户端浏览方式的统计信息。

17.5.4 转码与静态预生成的对比

转码的替代做法是在 Web 服务器上建立 Web 页面的不同副本，例如一个是 HTML，一个是 WML；一个图像分辨率高，一个图像分辨率低；一个有多媒体内容，一个没有。但是，这种方法不是很切合实际，原因很多：某个页面中的任何小改动都会牵扯很多页面，需要很多空间来存储各页面的不同版本，而且使页面编目和 Web 服务器编程（以提供正确的版本）变得更加困难。有些转码操作，比如广告插入（尤其是定向广告插入），就不能静态实现——因为插入什么广告和请求页面的用户有关。

对单一的根页面进行即时转换，是比静态的预生成更容易的解决方案。但这样会在提供内容时增加时延。不过有时候其中一些计算可以由第三方进行，这样就减少了 Web 服务器上的计算负荷——比如可以由代理或缓存中的外部 Agent 完成转换。图 17-3 显示了在代理缓存中进行的转码。

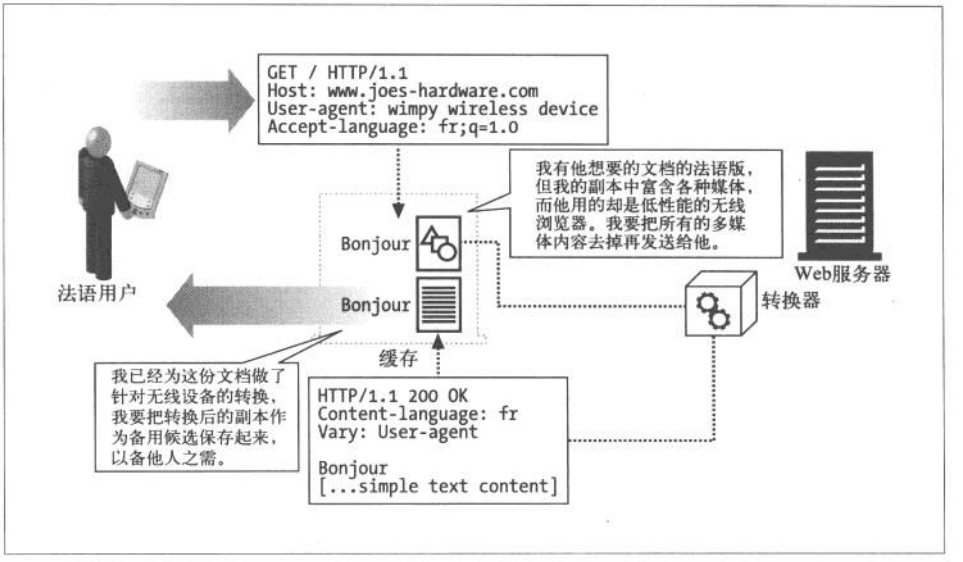

图 17-3 在代理缓存中进行转换或转码

17.6 下一步计划

由于以下两个原因，内容协商这个话题不只限于 Accept 和 Content 这两个首部集。

- HTTP 中的内容协商受到一些性能方面的限制。在各种变体中搜索合适的内容，或尽力“猜测”最佳匹配，都会有很大开销。有没有什么办法能专注内容协商协议以使这个过程更高效？ RFC 2295 和 RFC2296 尝试着对这个问题进行了研究，以提供透明的 HTTP 内容协商。

405

- HTTP 不是唯一需要进行内容协商的协议。在其他一些情况下，客户端也需要和服务器交互以便获得对客户端请求来说最好的答案，流媒体和传真就是另外两个例子。能否在 TCP/IP 应用层协议之上开发出通用的内容协商协议呢？内容协商工作组（Content Negotiation Working Group）就是专门为这个问题而成立的。这个工作组目前已经停止工作了，不过它提出了若干个 RFC。在下一节中，我们给出了这个组的网站链接。

17.7 更多信息

从下面的因特网草案和在线文档中可以获得更多内容协商方面的信息。

- http://www.ietf.org/rfc/rfc2616.txt
  RFC 2616，“Hypertext Transfer Protocol–HTTP/1.1”（“超文本传输协议 HTTP/1.1”），这是 HTTP 协议的当前版本，也是 HTTP/1.1 的官方规范。这份规范行文流畅、组织良好，是份详实的 HTTP 参考文献。不过对那些希望学习 HTTP 背后的各种概念和决策动机、弄清理论与实践不同之处的读者来说，它就不是很理想了。我们希望本书能补足背后的这些概念，使读者能更好地利用这份规范。 406

- http://www.ietf.org/rfc/rfc2295.txt
  RFC 2295，“Transparent Content Negotiation in HTTP”（“HTTP 中的透明内容协商”），这是一份备忘录，描述了建立在 HTTP 之上的透明内容协商协议。这份备忘录目前还是实验性的。

- http://www.ietf.org/rfc/rfc2296.txt
  RFC 2296，“HTTP Remote Variant Selection Algorithm RVSA 1.0”（“HTTP 远程变体选择算法 RVSA1.0”），这份备忘录描述了为特定的 HTTP 请求透明地选择“最佳”内容的算法。这份备忘录目前还是实验性的。

- http://www.ietf.org/rfc/rfc2936.txt
  RFC 2936，“HTTP MIME Type Handler Detection”（“HTTP MIME 类型处理器检测”），这份备忘录描述了一种用来判定浏览器支持的 MIME 类型处理器的方法。如果 Accept 首部不够明确的话，这种方法就能派上用场。

- http://www.imc.org/ietf-medfree/index.htm
  这个链接指向内容协商（简称 CONNEG）工作组网站。该工作组专注于 HTTP、传真和打印方面的透明内容协商。这个工作组目前已停止工作。 407

第五部分

内容发布与分发

第五部分讲述了 Web 内容发布和传播的各种技术。

- 第 18 章介绍了在现代的 Web 托管环境中部署服务器的若干方法，HTTP 对虚拟 Web 托管的支持以及如何在地理上相距遥远的服务器之间复制内容。
- 第 19 章讨论了创建 Web 内容并将其放置到 Web 服务器上去的各种技术。
- 第 20 章探讨了各种将来访的 Web 流量分发到一组服务器上的技术和工具。
- 第 21 章解释了日志的各种格式和各种常见问题。

408 ~ 410