其次，该 Web 服务器向缓存器发送具有被请求的对象的响应报文：

```
HTTP/1.1 200 OK
Date: Sat, 8 Oct 2011 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Wed, 7 Sep 2011 09:23:24
Content-Type: image/gif

(data data data data data ...)
```

该缓存器在将对象转发到请求的浏览器的同时，也在本地缓存了该对象。重要的是，缓存器在存储该对象时也存储了最后修改日期。第三，一个星期后，另一个用户经过该缓存器请求同一个对象，该对象仍在这个缓存器中。由于在过去的一个星期中位于 Web 服务器上的该对象可能已经被修改了，该缓存器通过发送一个条件 GET 执行最新检查。具体说来，该缓存器发送：

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-Modified-Since: Wed, 7 Sep 2011 09:23:24
```

值得注意的是 If-Modified-Since：首部行的值正好等于一星期前服务器发送的响应报文中的 Last-Modified：首部行的值。该条件 GET 报文告诉服务器，仅当自指定日期之后该对象被修改过，才发送该对象。假设该对象自 2011 年 9 月 7 日 09：23：24 后没有被修改。接下来的第四步，Web 服务器向该缓存器发送一个响应报文：

```
HTTP/1.1 304 Not Modified
Date: Sat, 15 Oct 2011 15:39:29
Server: Apache/1.3.0 (Unix)

(empty entity body)
```

我们看到，作为对该条件 GET 方法的响应，该 Web 服务器仍发送一个响应报文，但并没有在该响应报文中包含所请求的对象。包含该对象只会浪费带宽，并增加用户感受到的响应时间，特别是如果该对象很大的时候更是如此。值得注意的是在最后的响应报文中，状态行中为 304 Not Modified，它告诉缓存器可以使用该对象，能向请求的浏览器转发它（该代理缓存器）缓存的该对象副本。

我们现在完成了对 HTTP 的讨论，这是我们详细学习的第一个因特网协议（应用层协议）。我们已经学习了 HTTP 报文的格式，学习了当发送和接收这些报文时 Web 客户和服务器所采取的动作。我们还学习了一点 Web 应用程序基础设施，包括缓存、cookie 和后端数据库，所有这些都以某种方式与 HTTP 协议有关。

2.3 文件传输协议：FTP

在一个典型的 FTP 会话中，用户坐在一台主机（本地主机）前面，向一台远程主机传输（或接收来自远程主机的）文件。为使用户能访问它的远程账户，用户必须提供一个用户标识和口令。在提供了这种授权信息后，用户就能从本地文件系统向远程主机文件系统传送文件，反之亦然。如图 2-14 所示，用户通过一个 FTP 用户代理与 FTP 交互。该用户首先提供远程主机的主机名，使本地主机的 FTP 客户进程建立一个到远程主机 FTP 服务器进程的 TCP 连接。该用户接着提供用户标识和口令，作为 FTP 命令的一部分在该 TCP 连接上传送。一旦该服务器向该用户授权，用户可以将存放在本地文件系统中的一个或者多个文件复制到远程文件系统（反之亦然）。