器再到缓存的时延高估了从服务器到缓存的时延，但它是保守的。如果出错了，它只会使文档看起来比实际使用期要老，并引发不必要的再验证。计算是这样进行的： [191]

```
$apparent_age = max(0, $time_got_response - $Date_header_value);
$corrected_apparent_age = max($apparent_age, $Age_header_value);
$response_delay_estimate = ($time_got_response - $time_issued_request);
$age_when_document_arrived_at_our_cache =
    $corrected_apparent_age + $response_delay_estimate;
```

7.11.3 完整的使用期计算算法

上一节说明了当 HTTP 所承载的文档抵达缓存时，如何计算其使用期。只要将这条响应存储到缓存中去，它就会进一步老化。当对缓存中文档的请求到达时，我们需要知道文档在缓存中停留了多长的时间，这样才能计算文档现在的使用期：

```
$age = $age_when_document_arrived_at_our_cache +
    $how_long_copy_has_been_in_our_cache;
```

嗒嗒！这样就有了例 7-1 中给出的完整的 HTTP/1.1 使用期计算算法。这就是简单的簿记问题了——我们知道了文档是什么时候到达缓存的（$time_got_reponse），也知道当前请求是什么时候到达的（刚才），这样停留时间就是两者之差了。所有这些都以图形方式显示在图 7-18 中了。

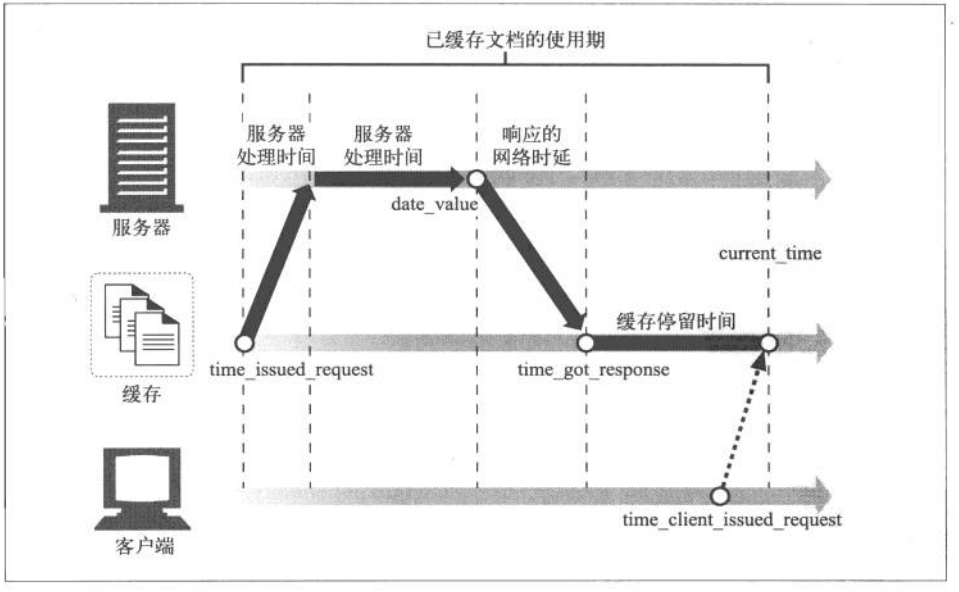

图 7-18 已缓存文档的使用期包括在网络和缓存中停留的时间