改变，则称为具有位置独立性（location independence）。将机器或服务器名称嵌在路径名中的分布式系统显然不具有位置独立性。一个基于远程安装（挂载）的系统当然也不具有位置独立性，因为在把某个文件从一个文件组（安装单元）移到另一个文件组时，是不可能仍旧使用原来的路径名的。可见位置独立性是不容易实现的，但它是分布式系统所期望的一个属性。

这里把前面讨论过的内容加以简要的总结，在分布式系统中处理文件和目录命名的方式通常有以下三种：

1）机器 + 路径名，如/machine/path 或 machine:path。

2)将远程文件系统安装在本地文件层次中。

3)在所有的机器上看来都相同的单一名字空间。

前两种方式很容易实现，特别是作为将原本不是为分布式应用而设计的已有系统连接起来的方式时是这样。而第三种方式的实现则是困难的，并且需要仔细的设计，但是它能够减轻了程序员和用户的负担。

4. 文件共享的语义

当两个或多个用户共享同一个文件时，为了避免出现问题有必要精确地定义读和写的语义。在单处理器系统中，通常，语义是如下表述的，在一个read系统调用跟随一个write系统调用时，则read返回刚才写入的值，如图8-38a所示。类似地，当两个write连续出现，后跟随一个read时，则读出的值是后一个写操作所存入的值。实际上，系统强制所有的系统调用有序，并且所有的处理器都看到同样的顺序。我们将这种模型称为顺序一致性（sequential consistency）。

在分布式系统中，只要只有一个文件服务器而且客户机不缓存文件，那么顺序一致性是很容易实现的。所有的read和write直接发送到这个文件服务器上，而该服务器严格地按顺序执行它们。

不过，实际情况中，如果所有的文件请求都必须送到单台文件服务器上处理，那么这个分布式系统的性能往往会很糟糕。这个问题可以用如下方式来解决，即让客户机在其私有的高速缓存中保留经常使用文件的本地副本。但是，如果客户机 1修改了在本地高速缓存中的文件，而紧接着客户机 2从服务器上读取该文件，那么客户机2就会得到一个已经过时的文件，如图8-38b所示。

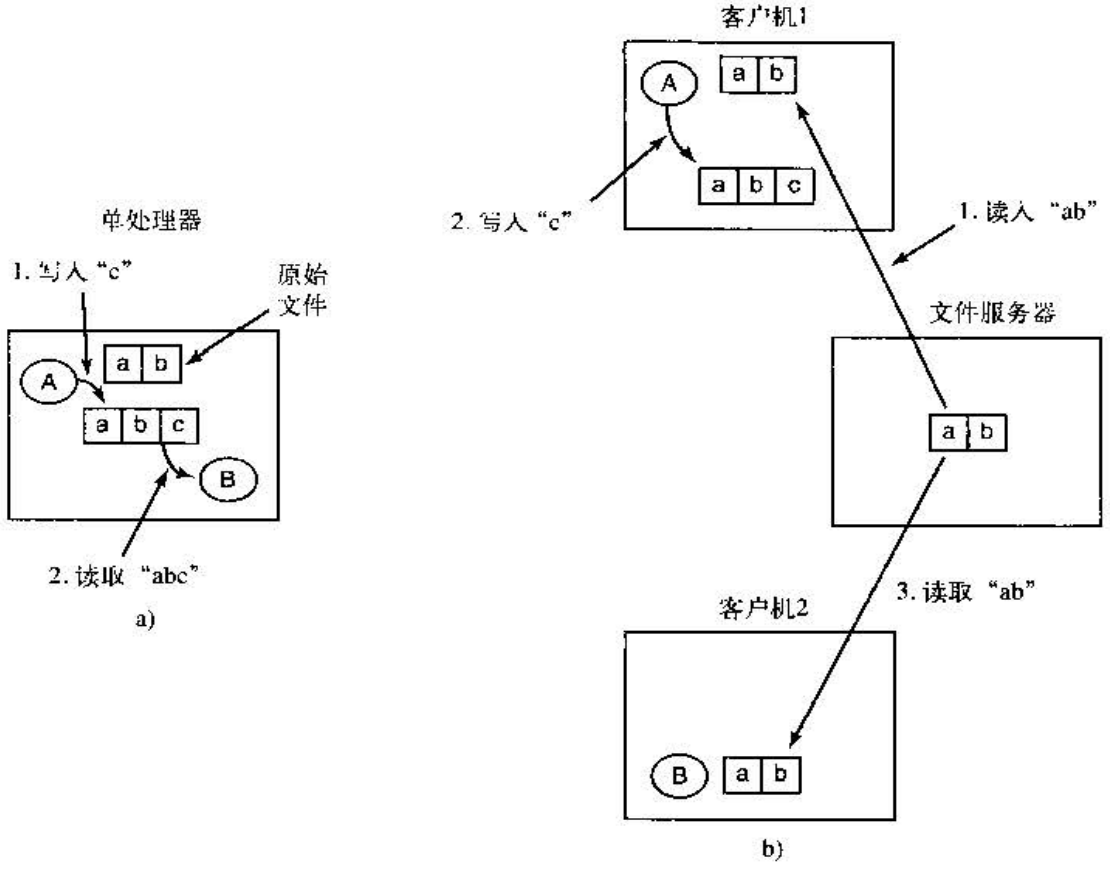

图8-38 a) 顺序一致性；b) 在一个带有高速缓存的分布式系统中，读文件可能会返回一个废弃的值