至多一次消息传递

第一个例子是如何强迫至多一次将消息传递到服务器上，即使系统面临崩溃也是如此。传统的方法是每条消息都带有唯一的消息号，每个服务器存储它所见过的所有消息号，这样它就可以检测出是新消息还是重发消息。此算法的问题是如果服务器故障重启，它将丢失消息号信息表。同样，应该每隔多长时间将消息号存储一次？

利用时间，可以把算法修改如下：每条消息都携带一个连接标识（由发送者选择）和时间戳，每次连接时，服务器记录它所见到的最近一次的时间戳，如果接收到的要求连接的消息其时间戳小于服务器记录的时间戳，则认为这条消息是已复制过的而被拒绝。可以移走旧的时间戳，但每台服务器上要一直保存一个全局变量：

G=当前时间-最大生存时间-最大时钟偏移

这里最大生存时间指一条消息能够存在的最长时间，最大时钟偏移指允许时钟与 UTC 偏移的最大值，任何比 G 早的时间戳都能安全地从表中移走，即所有老的消息都已消失了。如果接收到的消息是一个不可知的连接标识符，且它的时间戳比表中的新，则接收该消息，反之则拒绝。实际上，G 是所有老消息的消息数目的梗概。每隔一定时间 ΔT，将当前的时间写入磁盘。

当服务器故障重启时，从磁盘上重新装载 G，再加上⊿T。这样任何时戳老于 G 的消息因是复制过的而被拒绝接收。结果可能在故障前接收的消息在出现故障后就被拒绝了。而一些新的消息，也许会被不正确地拒收，但是在所有情况下，算法都保持了至多一次的思想。

基于时钟的缓冲存储器的一致性

第二个例子是考虑到在分布式文件系统中缓冲存储器的一致性。由于执行的原因，客户方能够本地缓冲文件是最理想的，但是，如果两个客户在同一时间修改同一文件会使缓存产生潜在的不一致性，通常的解决方案是区分为读缓存文件还是为写缓存文件。此方案的不足之处是如果一个客户有一个为读缓存的文件，在另一个客户得到为写而缓存的拷贝前，服务器必须先通知读方，使其拷贝无效，即使这个拷贝是几小时前做的。这种额外开销只需使用几个同步时钟即可完成。基本方法是当客户端需要一个文件时，给它一个标注有拷贝有效期的租约，当租约期满时，客户可以续租。租约到期时，就不能再使用缓存的拷贝。使用这种方法，当客户需要立即读文件时，它就可以申请。当租约使用期满时，仅仅是使用时间到，不需要发送消息告诉服务器，此时拷贝已经从缓冲存储器中删除了。如果租约已经到期，仍然想用缓冲存储器中的文件，客户可询问服务器，该文件的拷贝（标有时戳）是否仍然存在，若是，生成新的租约，但不需再传输文件。

如果一个或多个客户有为读而缓冲的文件，而另一个客户要写该文件，服务器将不得不请求读方提前终止其租约。若一个或多个客户方有故障，服务器将等待一定时间，至到服务器租约到期，在传统算法中，要求从客户方到服务器方必须有一个明确的“允许使用缓冲存储器”的答复，如果服务器要求客户返回文件，或客户已返回文件而对方无反应，意味出现

了故障。这时服务器不知道是客户方崩溃了还是反映太慢，为了解决这个问题，使用基于时钟的算法，服务器等待到租约到期为止。