OT（Operation Transformation），操作转换，协同技术中用来保持不同的数据副本一致性的一种方法。在不同的终端，根据操作顺序的不同，对操作进行调整，以保持数据一致性。

实时协同编辑，是指多人同时编辑一个文档，最典型的例子是 Google Docs，你可以实时看到别人做出的修改，不用手动刷新页面。

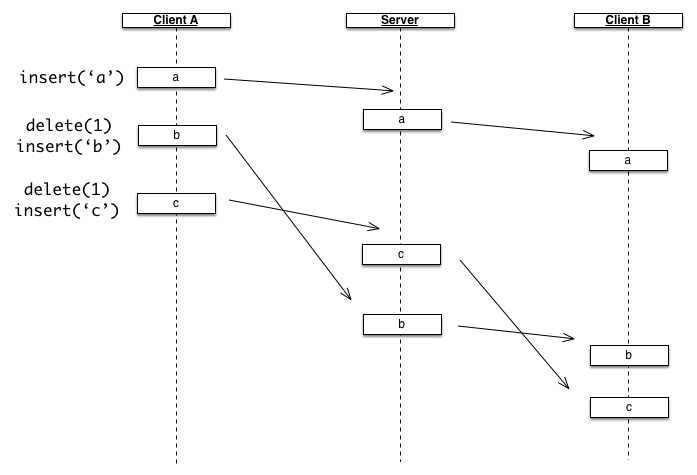
要实现实时编辑，我们需要解决两个技术点：实时通信问题、编辑冲突问题，其中实时通信问题比较好解决，可以使用 long pull 或 WebSocket，编辑冲突问题的解决方案有「编辑锁」、「GNU diff-patch」、「Myer’s diff-patch」、「Operational Transformation」和「分布式 Operational Transformation」。本文只讨论OT转换规则的实现和难点部分。

编辑锁，最为简单的一种方案。当用户编辑文档时给文档加上一个状态，在此状态下其他用户是无法编辑该文档的。根据产品需求可以做成全文档的编辑锁或者是段落锁，具体的难点在于对文本编辑器的改造。

OT，在OT规则下任意文本内容修改可以转成以下 3 种类型的操作(Operational)：

1. retain(n)：保持 n 个字符，也就是说这 n 个字符不变
2. insert(str)：插入字符 str
3. delete(str)：删除字符 str

当文本在不同用户间同步时，需要在服务端做一次针对全文本的OT转换，再将转换后的操作步骤发给相关用户，用户将服务器的转换步骤按同样的算法对本地文本做一次转换。这样实现文本的信息同步，保证用户a - 服务器 - 用户b之间的文本是一致的。



**基于分布式的OT转换规则**

问题小结，1：现有开源的基于分布式OT的文档协同系统所采用的编辑器是codeMirror，只能处理纯文本的协同（图片和链接的处理实际上是把原有文本替换成图片链接和超链接，在浏览器中会把链接显示成图片和超链接，列表样式是模拟的样式）。codeMirror是基于文本行，对于表格无法做到文本行方式的显示（对于单元格td的内容无法用第几行第几个字符位置来标识）

2：所有的合并操作放在服务端，当文本越来越大，参与协同的用户越来越多时，服务器的性能瓶颈影响协同的效率（遍历文本会浪费大量时间和内存）

3：分布式OT下，任意用户或服务器在转换操作步骤时出错后会浪费这次合并转换且会把这次的变更丢弃，不能实现真正意义上的协同（服务端的文本不是真正的全用户文档合并结果，用户的文档状态和结果是基于服务器文档来实现的）