

# 协同编辑系统设计

## Collaborative Editing System

主讲人 南帝

WebSocket  
FileSystem  
Redis  
Collaborative  
Mysql CRDT  
Domain OT  
Javascript  
Database

多人同时在线编辑同一份文档  
比如Google Docs

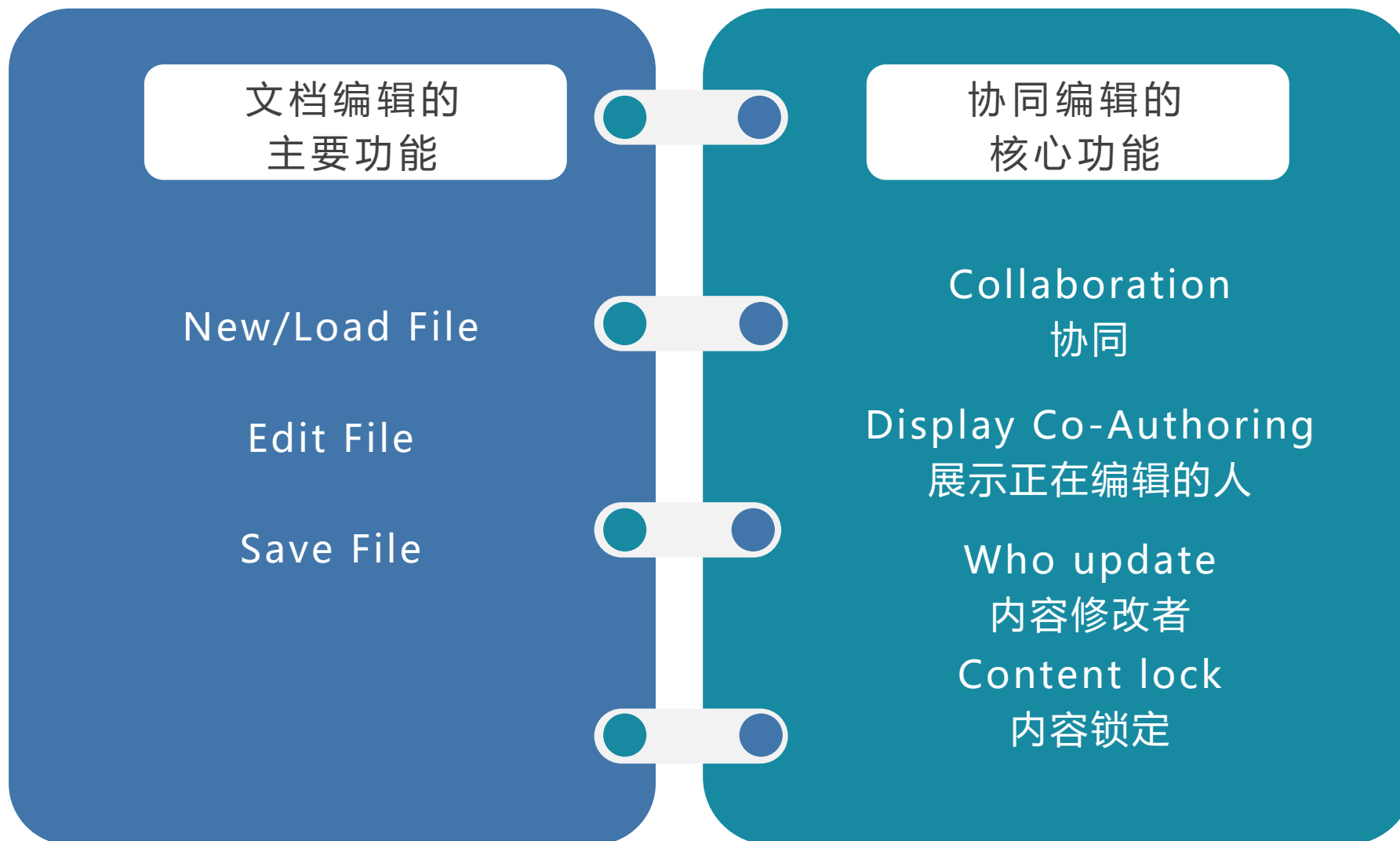


文档编辑的  
主要功能

New/Load File

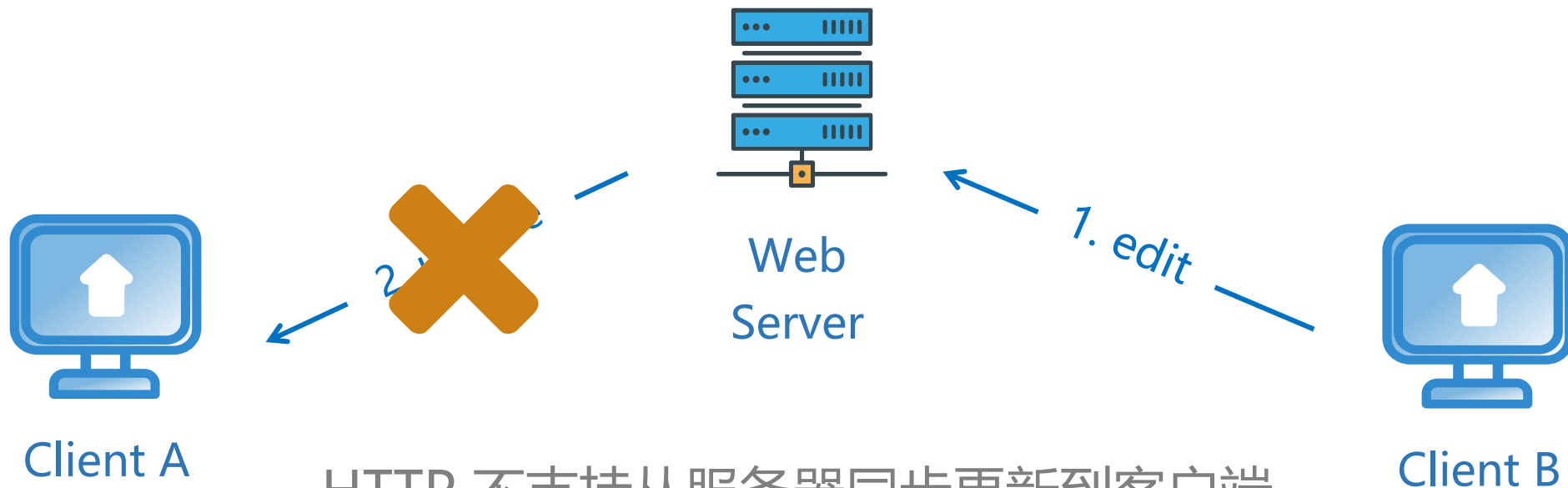
Edit File

Save File





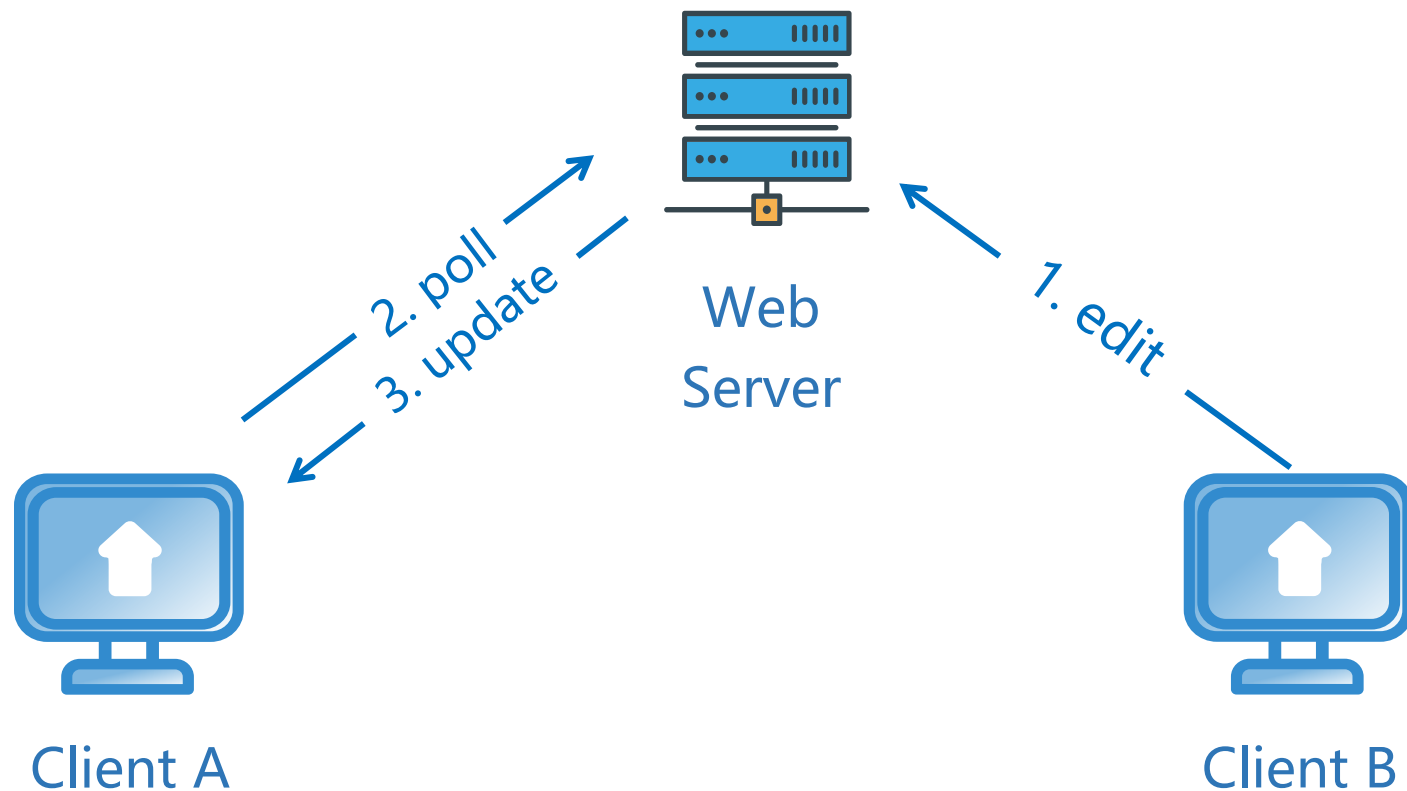
同步  
更新



HTTP 不支持从服务器同步更新到客户端

我们是不是不能用HTTP 来实现同步更新的操作?





这样会有什么问题?

- 1. 服务器压力大
- 2. 实时性(real-time)不高

01

## 链接无法复用，即不支持长链接(Long connection)

http 1.0 规定浏览器与服务器保持较短时间的链接，浏览器每次请求都和服务器经过三次握手(Three-way Handshake)和慢启动(Slow Start)，建立一个TCP链接，服务器完成请求处理后立即断开TCP链接。

当一次请求完成后，就断开了链接，如果要进行第二次的链接，第二次请求过来的时候需要重新再通过前面的链接过程，耗时

02

## 线头阻塞 Head of Line (HOL) Blocking

请求队列的第一个请求因为服务器正忙（或请求格式问题等其他原因），导致后面的请求被阻塞(block)。一个连接就是一个单线程在处理，服务器只能响应我的一次的request，后面的请求都会被阻塞。

### 存在的问题

早期的网络应用都是通过HTTP1.0的方式进行交互的，但是现在的网络应用变的比之前更加复杂，需要交互的场景变的很多，客户端和服务器的交互比较频繁；

通过HTTP1.0这种形式，客户端需要不断的发请求给服务器，服务端需要不断的去响应请求。这样对服务端和客户端的压力都比较大，而且交互的效率比较低。

01

## 支持长链接(Long connection)

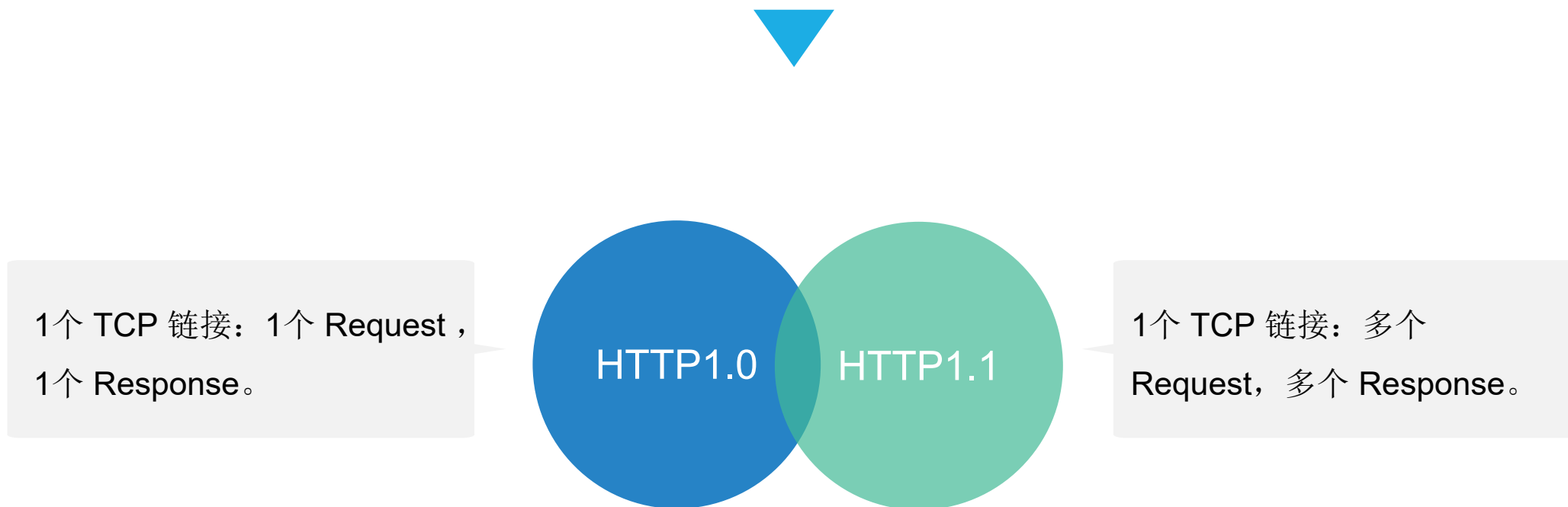
一个TCP链接可以传送多个http请求(Request)和响应(Response)，减少了TCP建立链接和关闭链接的消耗。

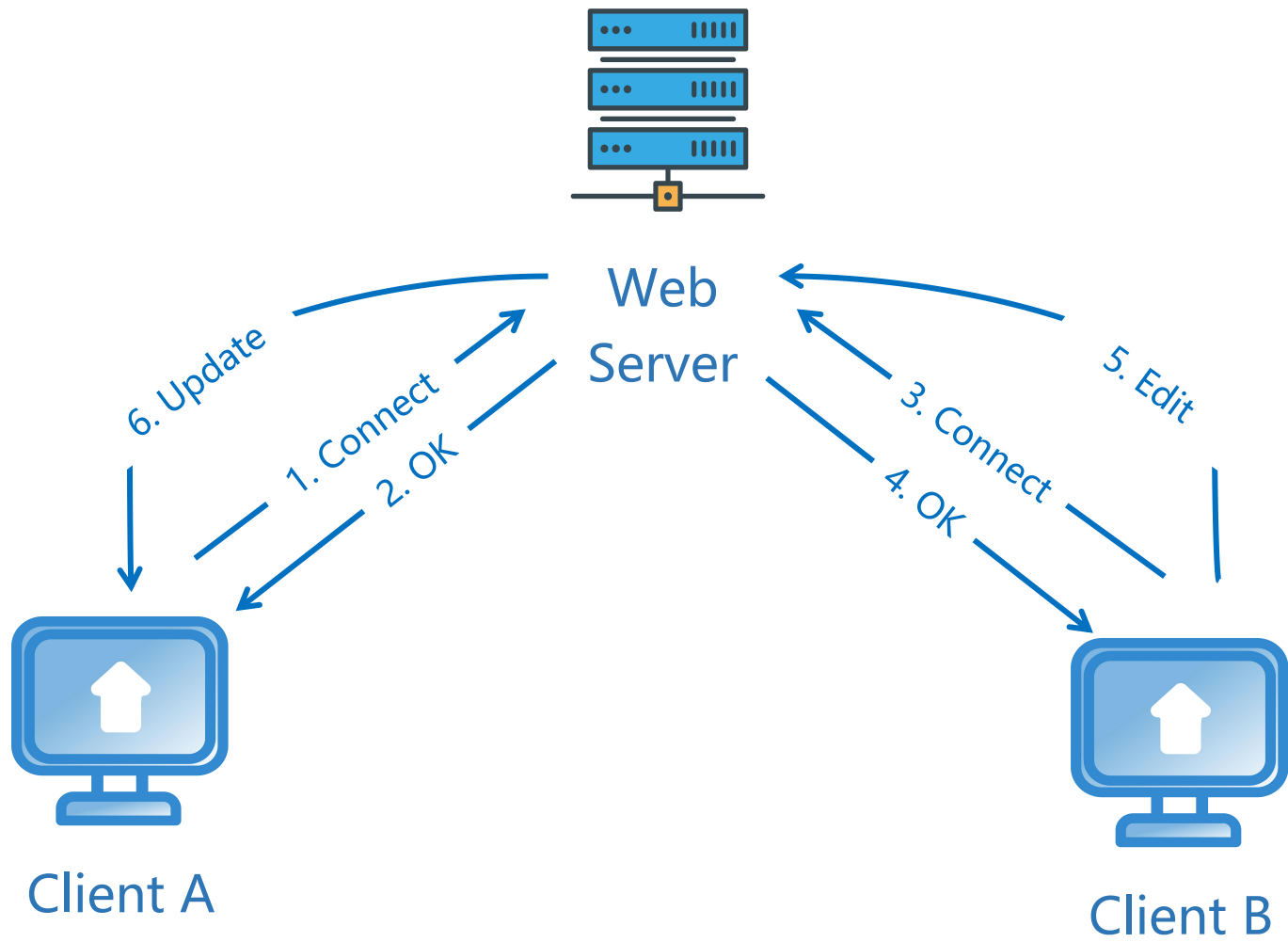
02

## 支持 HTTP 管道(Pipeline)

管道可以让我们把 FIFO 队列从客户端（请求队列 Request Queue）迁移到服务器（响应队列 Response Queue），即客户端可以并行(parallel)，服务端串行(Serial)。

http1.1允许客户端不用等待上一次请求结果返回，就可以发出下一次请求，但服务器端必须按照接收到客户端请求的先后顺序依次回送响应结果





1.先建立连接

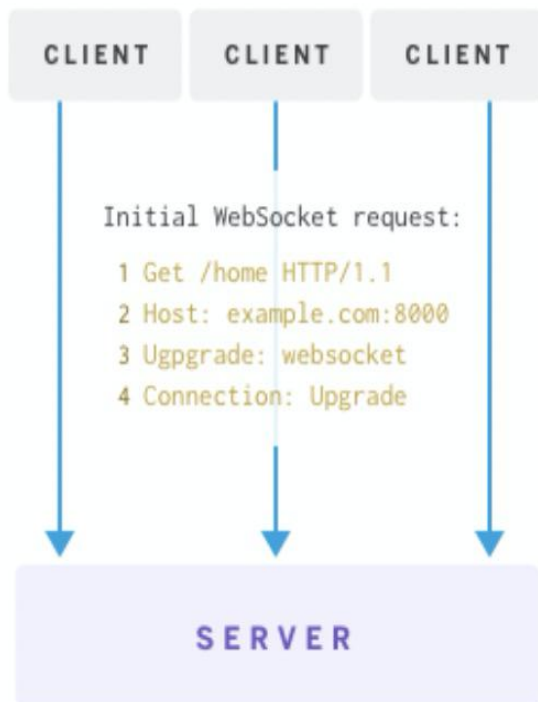
2.保持住会话(Session)

3.更改内容同步

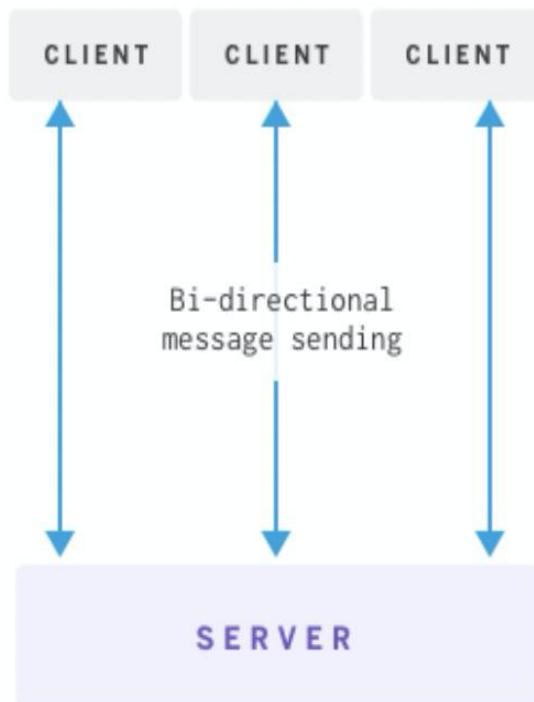
客户端的更改会同步到其他客户端，  
WebSocket 拿到了客户端A更改的内容后，  
会主动的将更改的内容推送给客户端B

## WEBSOCKET PROTOCOL

**Step 1:** A Request to initiate a WebSocket connection is sent to the server, from any number of clients



**Step 2:** Open and maintain WebSocket connections between multiple clients and a single server



## 解决的问题

服务端和客户端的交互的方式，WebSocket 能够双向通讯  
HTTP1.0 和 HTTP1.1 都不能都服务端给客户端发送东西，  
服务端只能被动的去接收。

## 特点

### 1.支持双向通讯

- 服务端可以主动的去给客户端发送一个 Response
- 客户端也可以发送请求给服务端

### 2.实时性强

服务端主动发请求给客户端，解决了实时性的问题

例：股票交易系统，页面会自动实时刷新股票价格信息

WebSocket 的出现，使浏览器具备实时双向通信的能力



请求  
文件

## 传统文件系统是如何查询文件的？

路径 + 文件名

如：http://xxx.com/folder\_1/folder\_2/...../file\_name

## 是否适用于 Web？

不适用



## 原因

### 1. 文件被分享时，路径中可能包含隐私信息

如：http://xxx.com/南帝/学习资料/苍老师.mp4

### 2. 链接太长不利于分享

参见短网址

## 如何设计 URL 更加合理？

为每一个文件生成一个唯一的 key，用于标识这个文件

如：http://xxx.com/[wwuSA1cs4qtS](#)

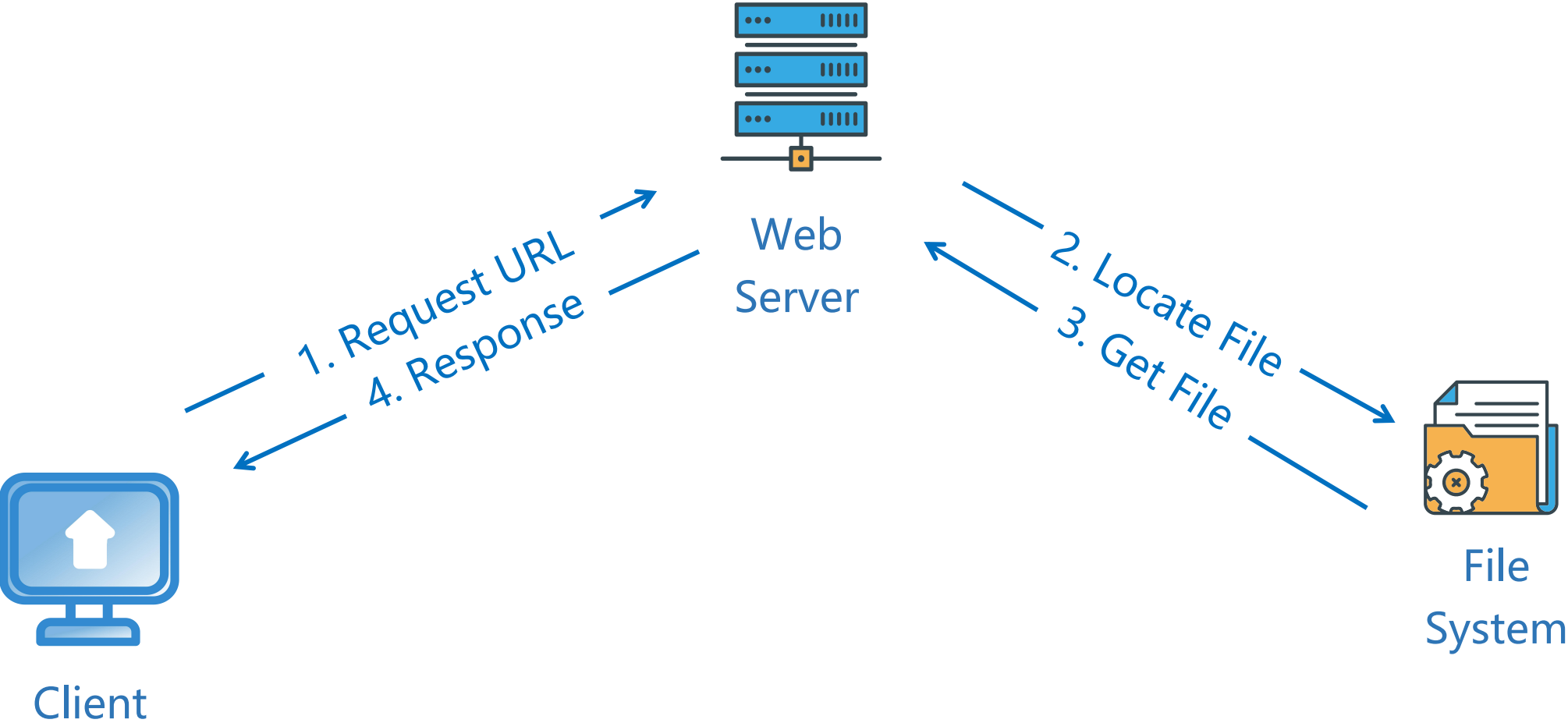
key 可以使用 uuid。


# 加上文件slugify字符串作为后缀

`https://xxx.com/wwuSA1cs4qtS/collaborative-editing`

URL 可读性强

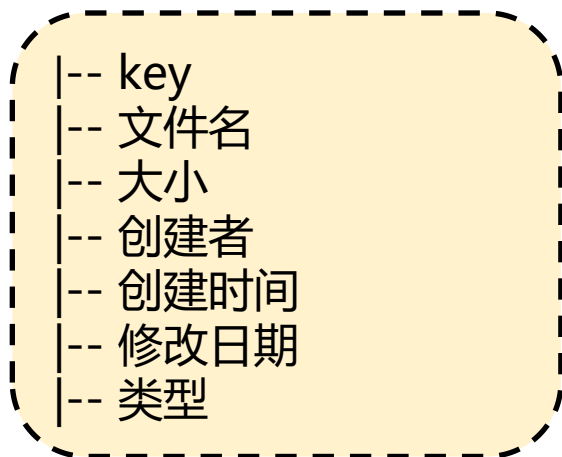
Slugify | the-online-slug-generator: <https://blog.tersmitten.nl/slugify/>





# 文件 存储

## Metadata



Metadata --> DB

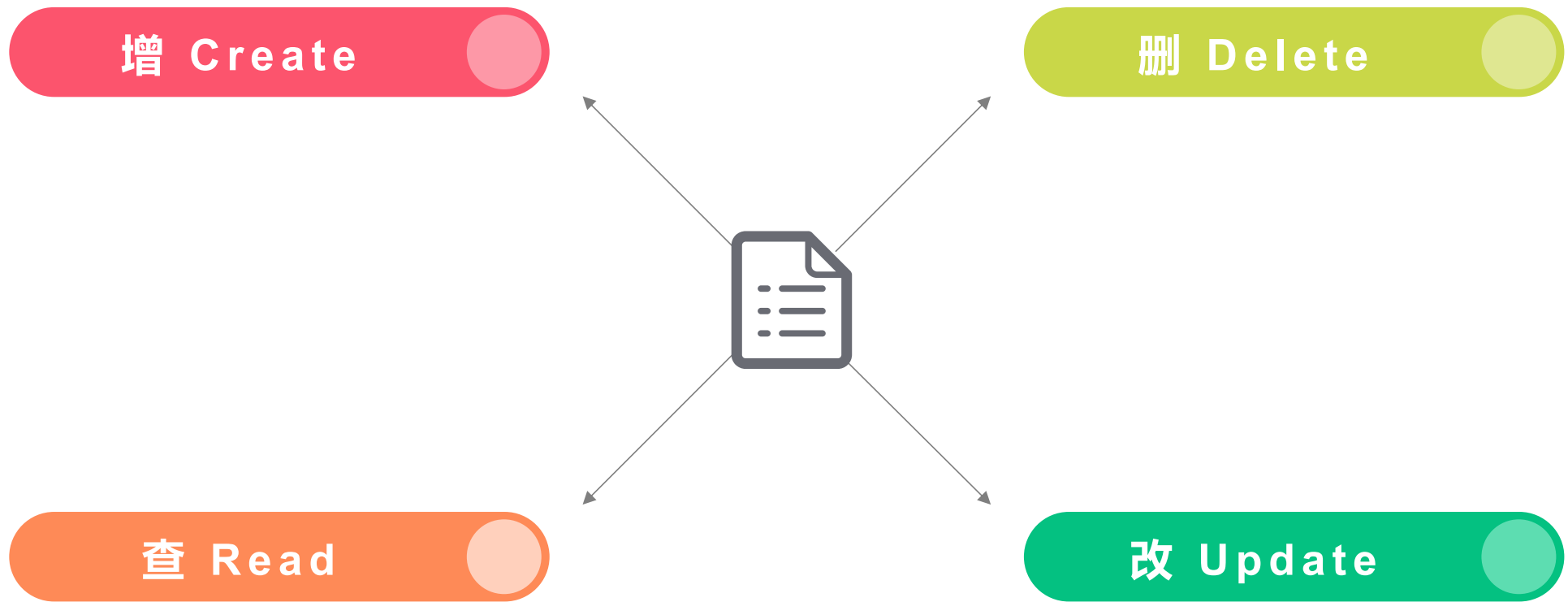
## Content



Content --> 文件系统

## 为什么 Content 不存 Metadata 的信息?

因为 Metadata 是一种结构化信息，适合存储在数据库里；而 Content 是一种文本信息，适合存储在文件里



如果用户以整个文件为粒度(Granularity)进行操作，会有什么问题？

操作之后会改动整个文件内容

## 优化方式

以“行”为颗粒度(Granularity), 如: Quip、飞书等

以“词”为颗粒度, 如: Google Doc、石墨等



为方便多个用户进行以行为单位的增删查改，文件中的内容应该如何优化？

A. Text

B. List of row text

C. LinkedHashMap

<https://www.lintcode.com/problem/lru-cache/description>

# 为何要用双向链表存放数据呢？

- 在文件服务器上是以具体的文件形式存储
- 应用服务器从文件服务器上 load 过来了，文件的内容是存储在内存中
- 链表的优势在任意位置插入或者删除，性能好
- 文件中每一行的数据匹配链表的一个节点
- 内容的变化相当于链表节点的变化

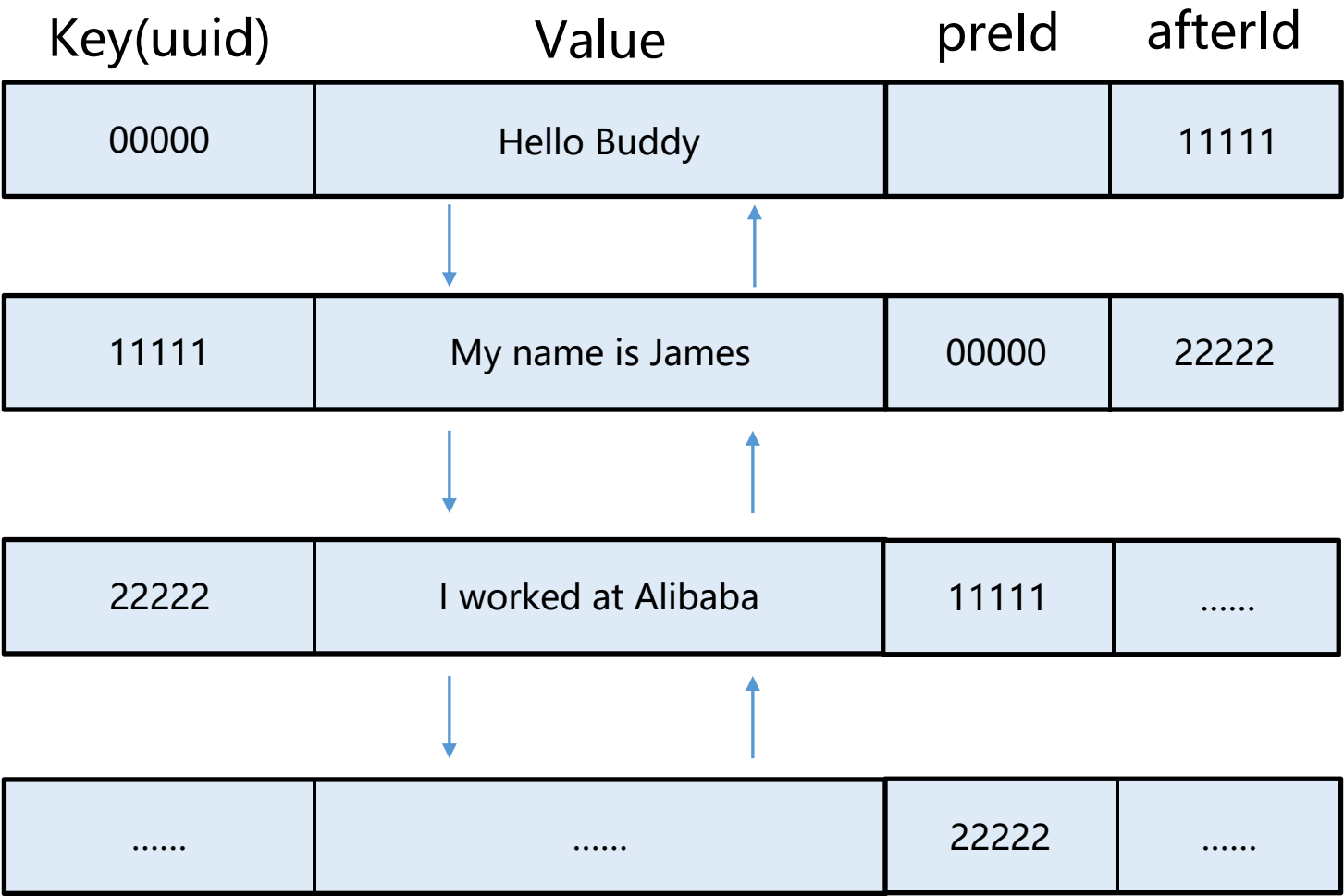
那文件内容怎么转换成链表的呢？

## Introduce.txt

Hello Buddy  
My name is James  
I worked at Alibaba  
...  
...  
...  
Bye!

每一行都是一个结构体  
Key: 文件每一行的唯一标识  
Value: 每一行的内容  
preId: 前驱Id, 前面一行  
afterId: 后续Id, 后面一行

唯一标识由前端生成  
在前端页面渲染时生成每一行的uuid, 并记录下来



那前端新增或者变更内容，需要传哪些信息给服务端呢？

- 修改/新增了哪一行
- 代表前后行的唯一标识
- 操作的时间
- 操作类型，新增/删除/修改

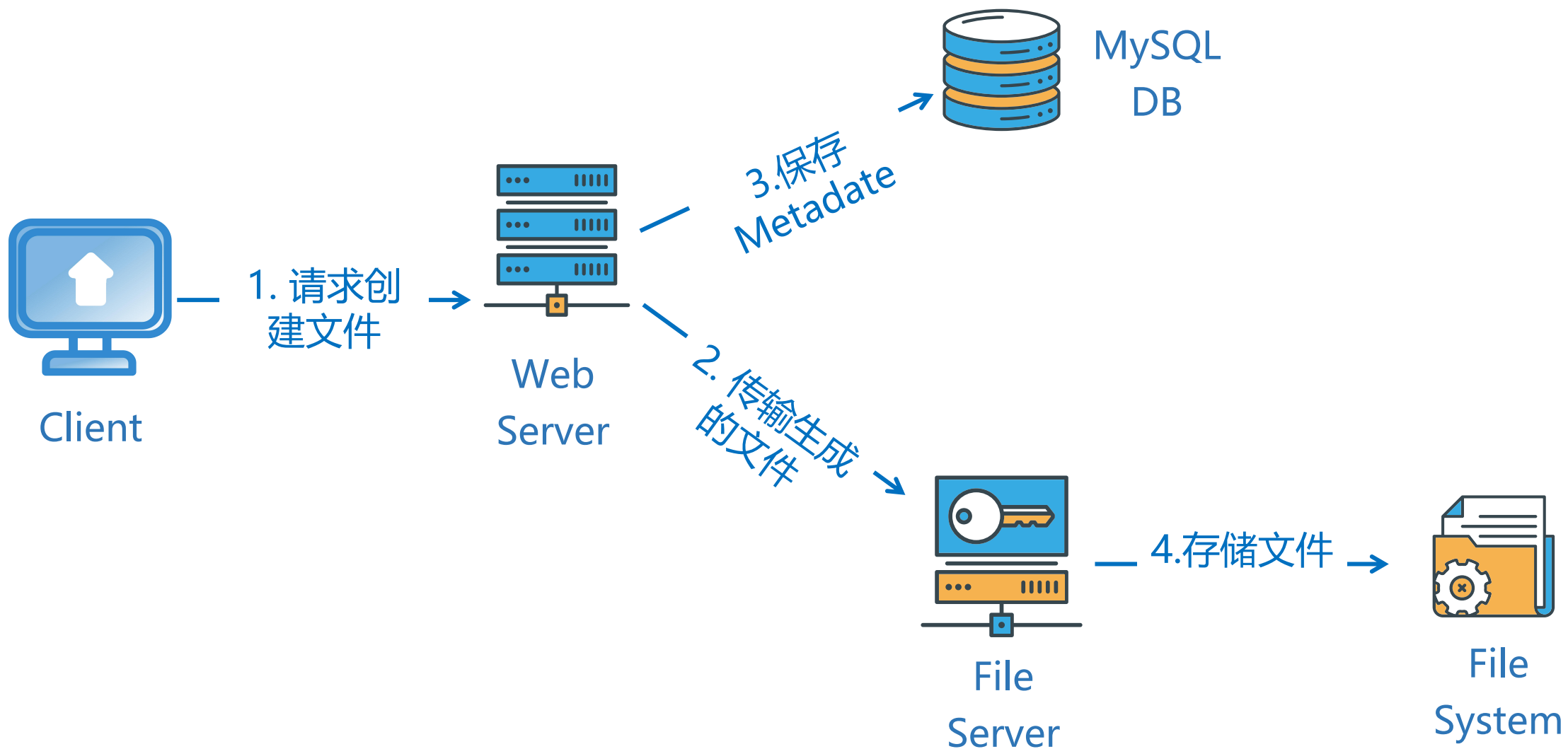
```
{
  "changes": [
    {
      "uid": "33333",
      "content": "updated content",
      "date": 1596044453949220,
      "preId": "00000",
      "afterId": "55555",
      "opType": "update"
    },
    {
      "uid": "44444",
      "content": "new content",
      "date": 1596064880092835,
      "preId": "33333",
      "afterId": "55555",
      "opType": "insert"
    }
  ]
}
```

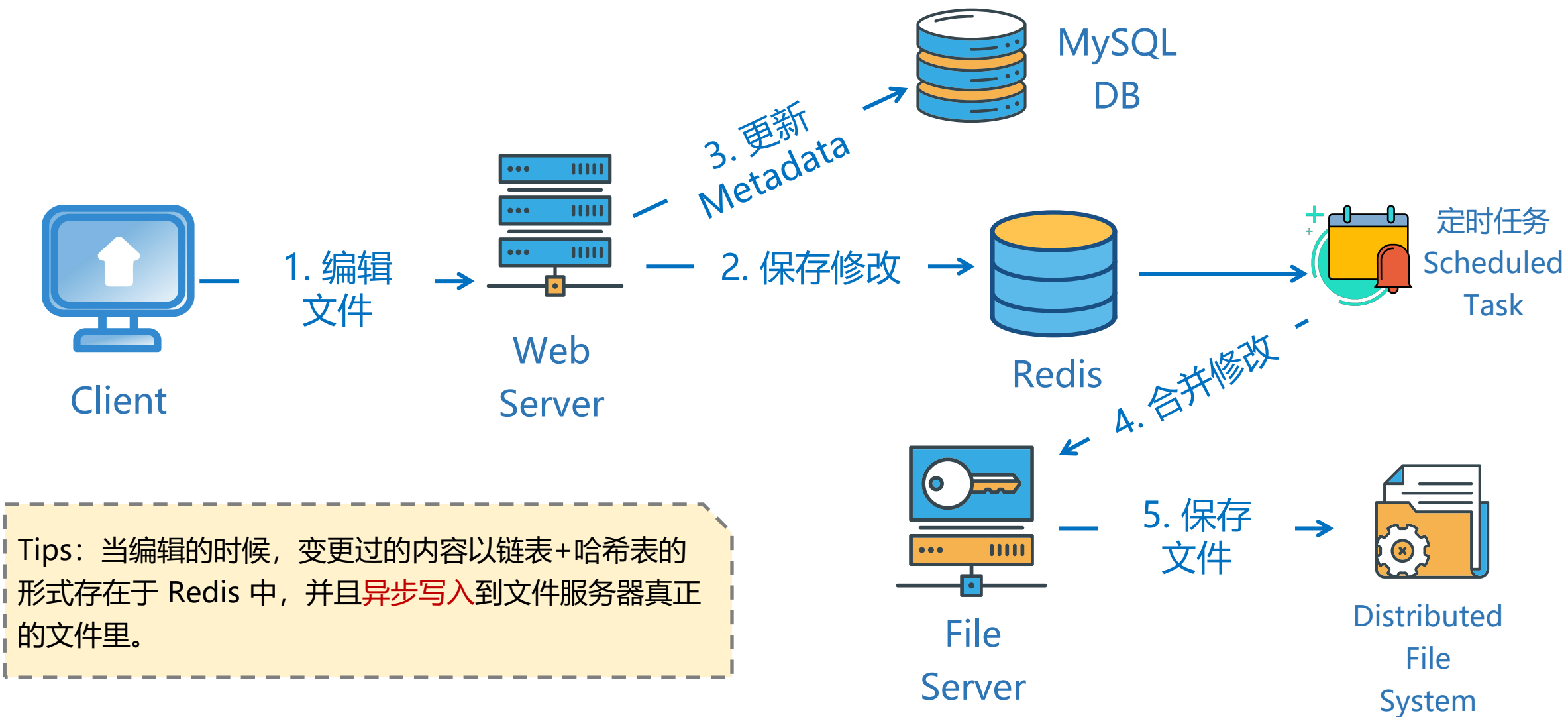
- 1.客户端告诉服务端，在什么时间点，在哪一行，做了什么操作，操作的内容是什么
- 2.服务端拿到前端的请求后，经过处理，同步给其他在编辑的客户端
- 3.其他客户端拿到服务端同步过来的信息后，会根据传过来的信息，将变化的内容在本地页面进行替换

JSON: JavaScript 对象表示法 (JavaScript Object Notation) 是一种轻量级的数据交换格式。它基于ECMAScript的一个子集。JSON采用完全独立于语言的文本格式, 但是也使用了类似于C语言家族的习惯 (包括C、C++、C#、Java、JavaScript、Perl、Python等) 。这些特性使JSON成为理想的数据交换语言。易于人阅读和编写, 同时也易于机器解析和生成(一般用于提升网络传输速率)。JSON 解析器和 JSON 库支持许多不同的编程语言。JSON 文本格式在语法上与创建 JavaScript 对象的代码相同。由于这种相似性, 无需解析器, JavaScript 程序能够使用内建的 eval() 函数, 用 JSON 数据来生成原生的 JavaScript 对象。JSON 是存储和交换文本信息的语法。类似 XML。JSON 比 XML 更小、更快, 更易解析。JSON 具有自我描述性, 语法简洁, 易于理解。

# 前端传到服务端之后做了哪些事情？







MySQL:

文件Table

文件表	文件ID	文件名称	文件URL	文件后缀名
	wwuSA1cs4qtS	memo	http://xxx/memo.txt	txt

Redis:

Key: file key  
Value: {  
    dummy: string;  
    tail: string;  
}

Key: file key + row key  
Value: {  
    content: string;  
    preID: string;  
    nextID: string;  
}

(00000) Hello Buddy  
(11111) My name is James  
(99999) **My favorite color is blue**  
(33333) I worked at Alibaba  
...  
...  
...  
(12345)Bye!

Before

11111	My name is James	00000	33333
33333	I worked at Alibaba	11111	44444

After

11111	My name is James	00000	99999
99999	<b>My favorite color is blue</b>	11111	33333
33333	I worked at Alibaba	99999	44444

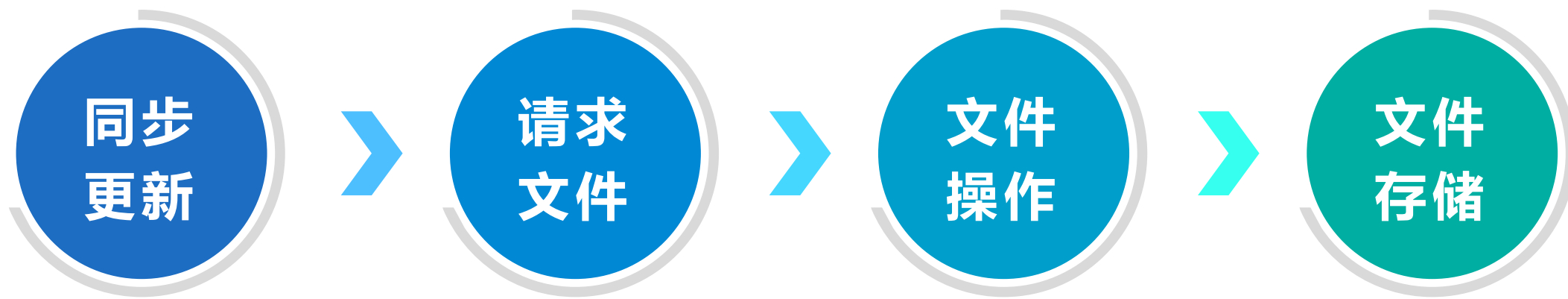
(00000) Hello Buddy  
(11111) My name is James **Taylor**  
(33333) I worked at Alibaba  
...  
...  
...  
(12345)Bye!

Before

11111	My name is James	00000	33333
33333	I worked at Alibaba	11111	44444

After

11111	My name is James <b>Taylor</b>	00000	33333
33333	I worked at Alibaba	11111	44444



## 如何在网页中显示有多个人同时编辑的头像？



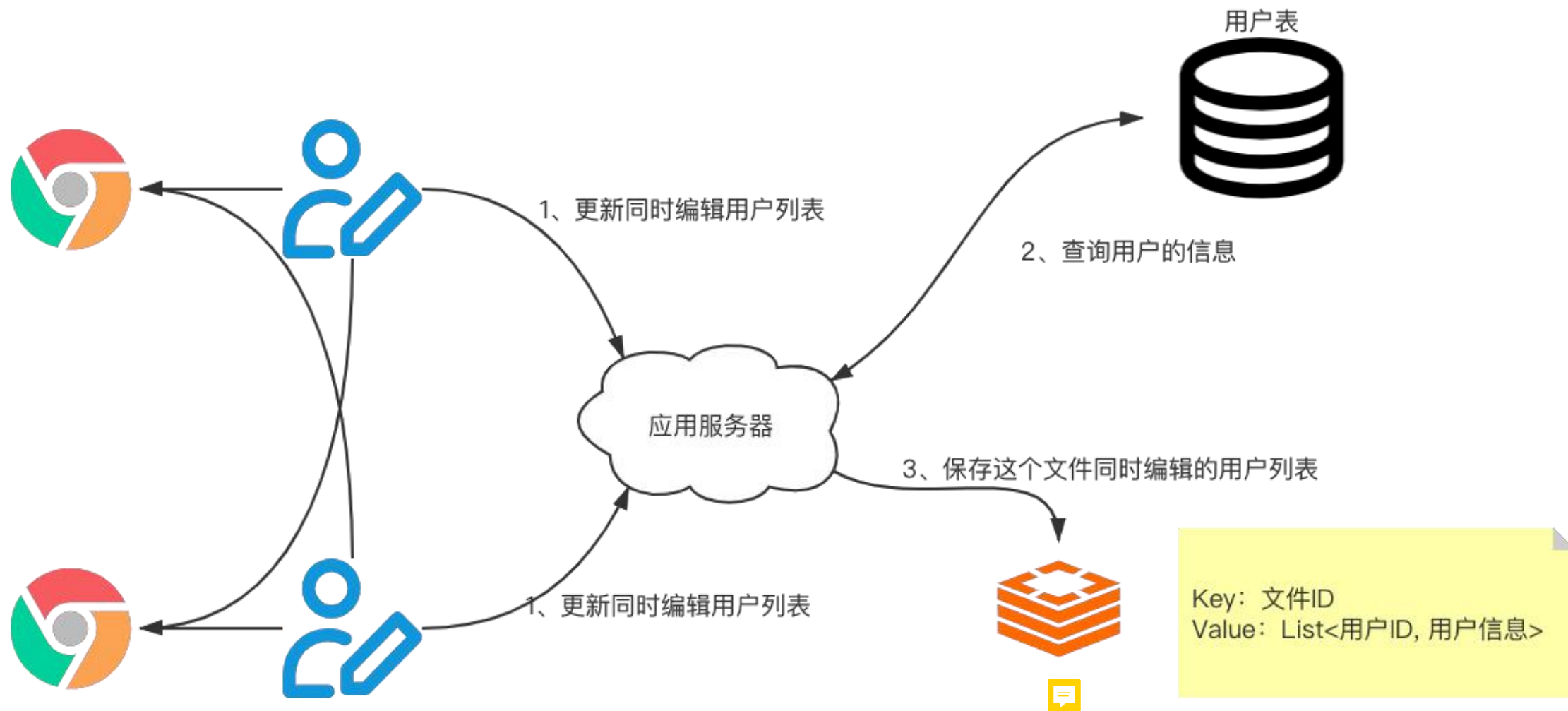
# 同时编辑一份文档的用户列表适合存在哪？

A. NoSQL

B. SQL

用户列表的数据信息是高频变化的





用户Table

用户表	用户ID	用户名	性别	头像URL
	12345	James	male	http://xxx/xx.png

## 如何记录某一行谁正在改？

**test**

hello world

I am here

where are you

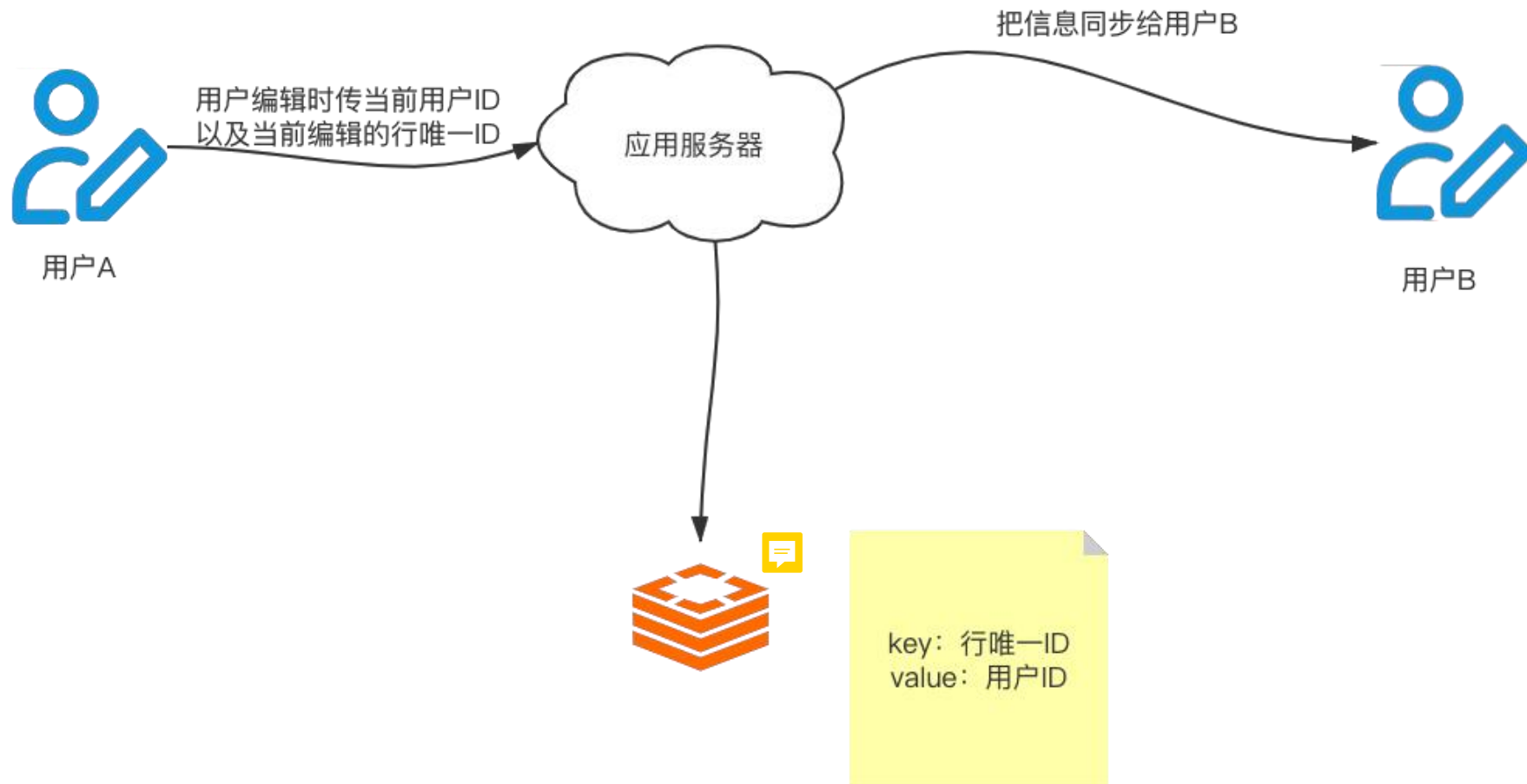
I am ok

来了 hello

正在编辑

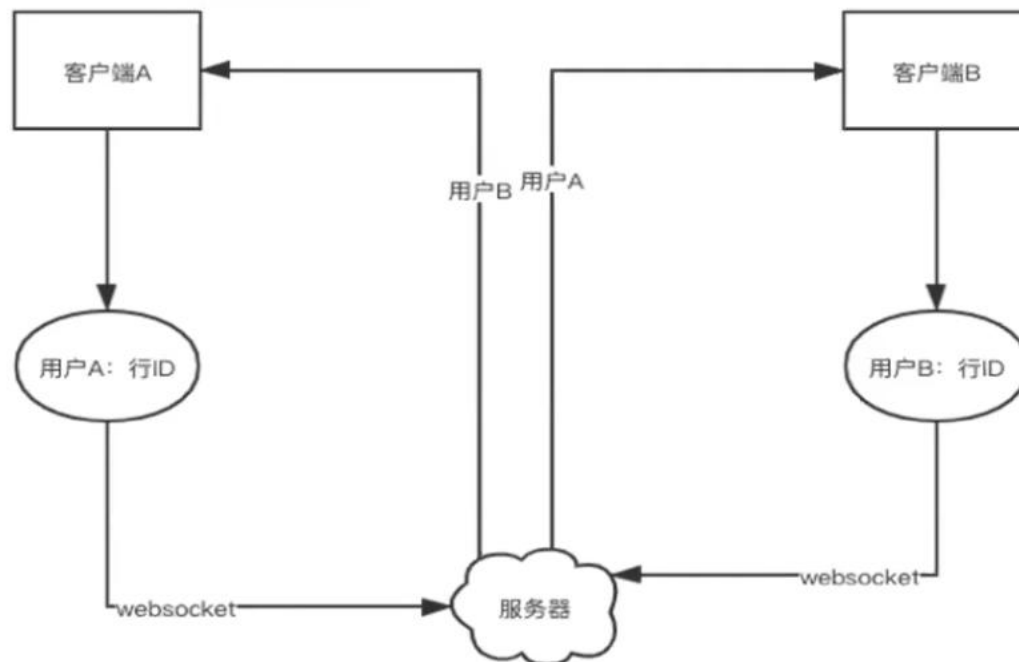


# 显示某一行内容正在被谁编辑



## 基于这个流程，行锁定如何实现？

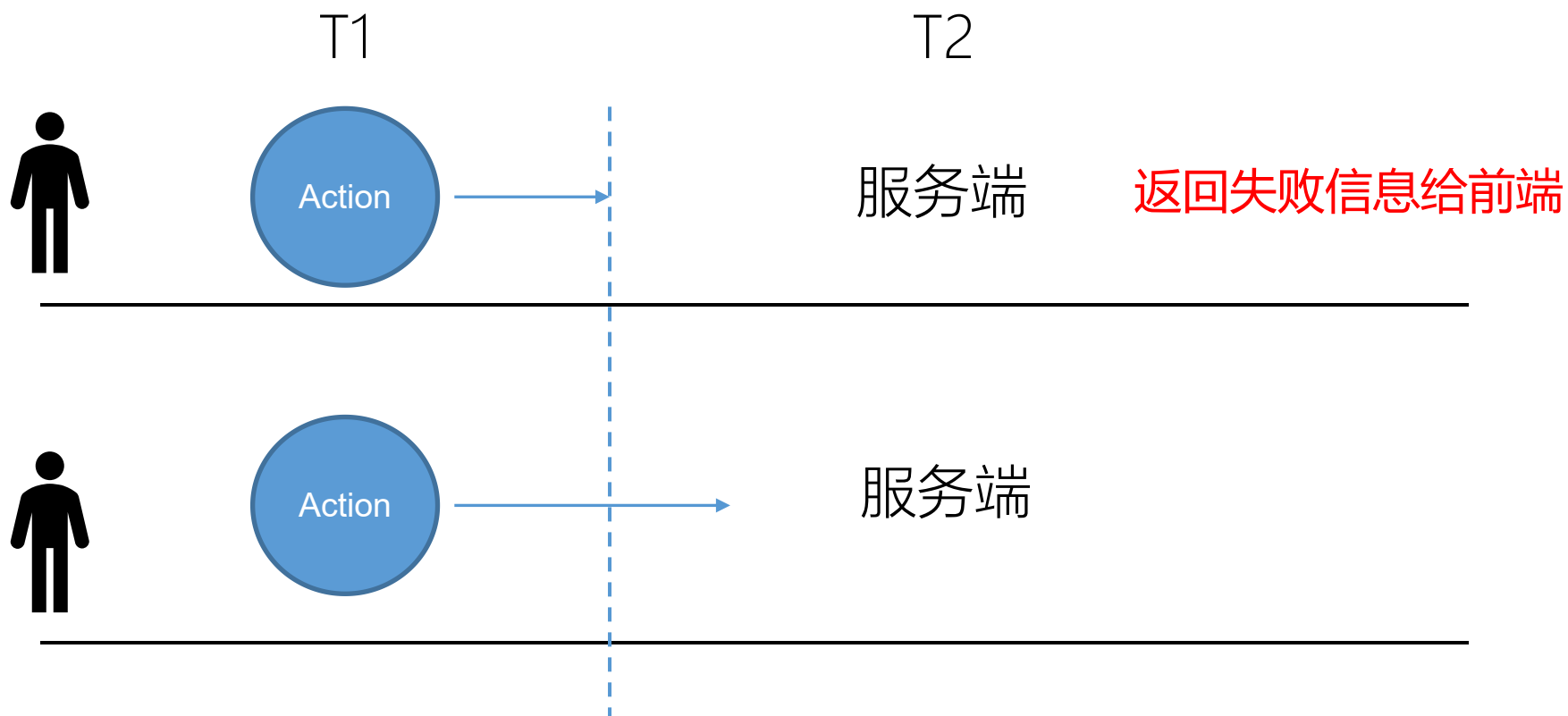
前端判断当前用户ID和正在编辑  
行的用户ID是否同一人



## 那多用户同时编辑，可能会遇到哪些Case?

- 多人同时编辑某一行（同时锁定）
- 一个人正在删除某一行内容，一个人正在增加内容

## 通过到达服务端的时间来决定采纳谁的



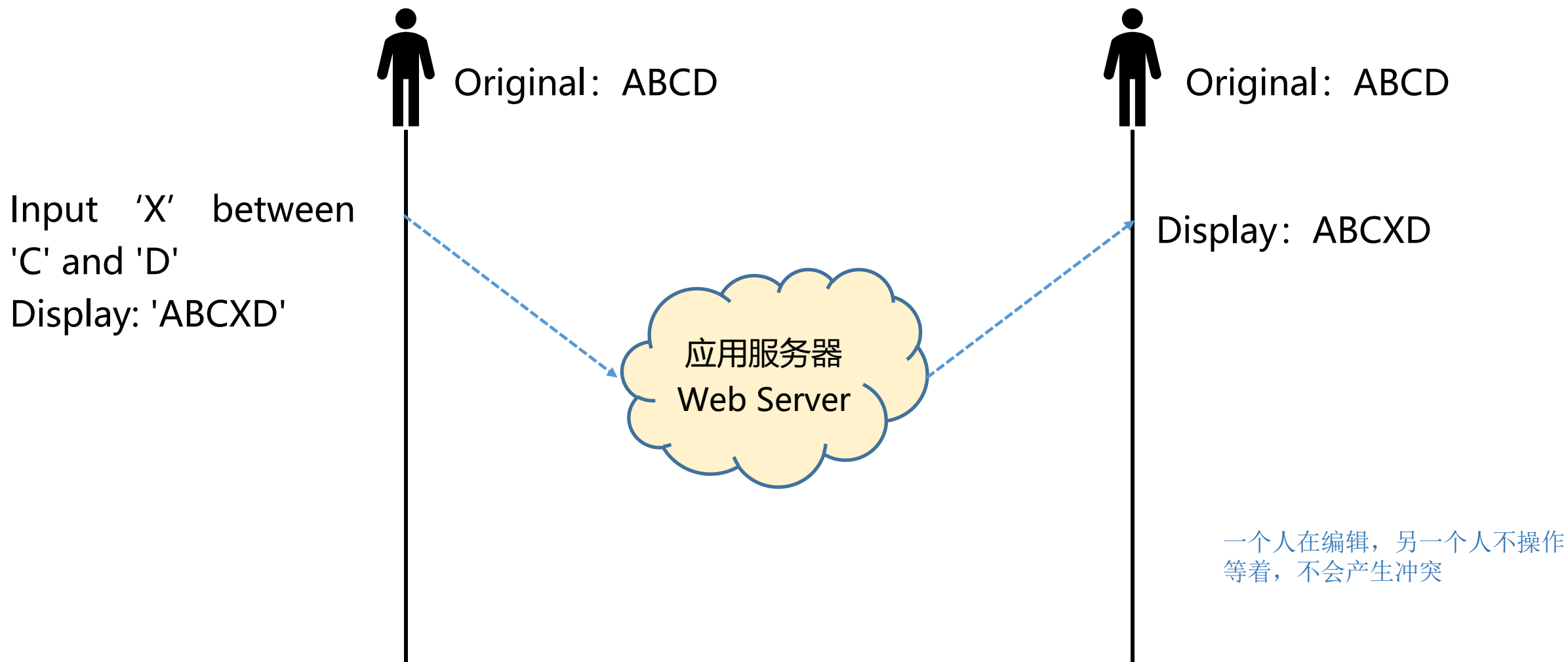
之前的设计都是基于行锁定的设计，有没有不需要锁定的？

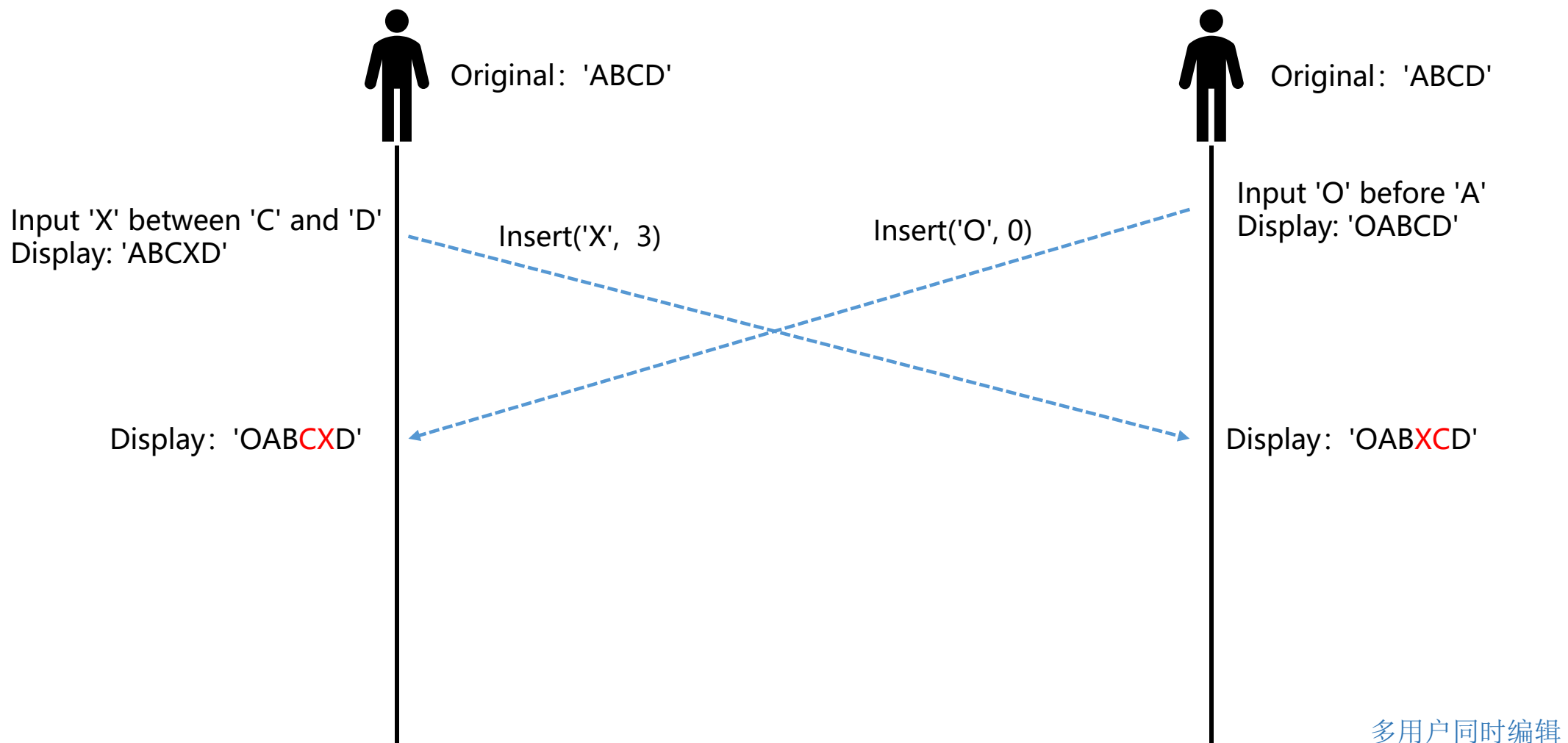
- OT(Operational Transform)

课程主要介绍OT算法设计思想，不涉及具体的实现  
Google Docs 基于OT算法

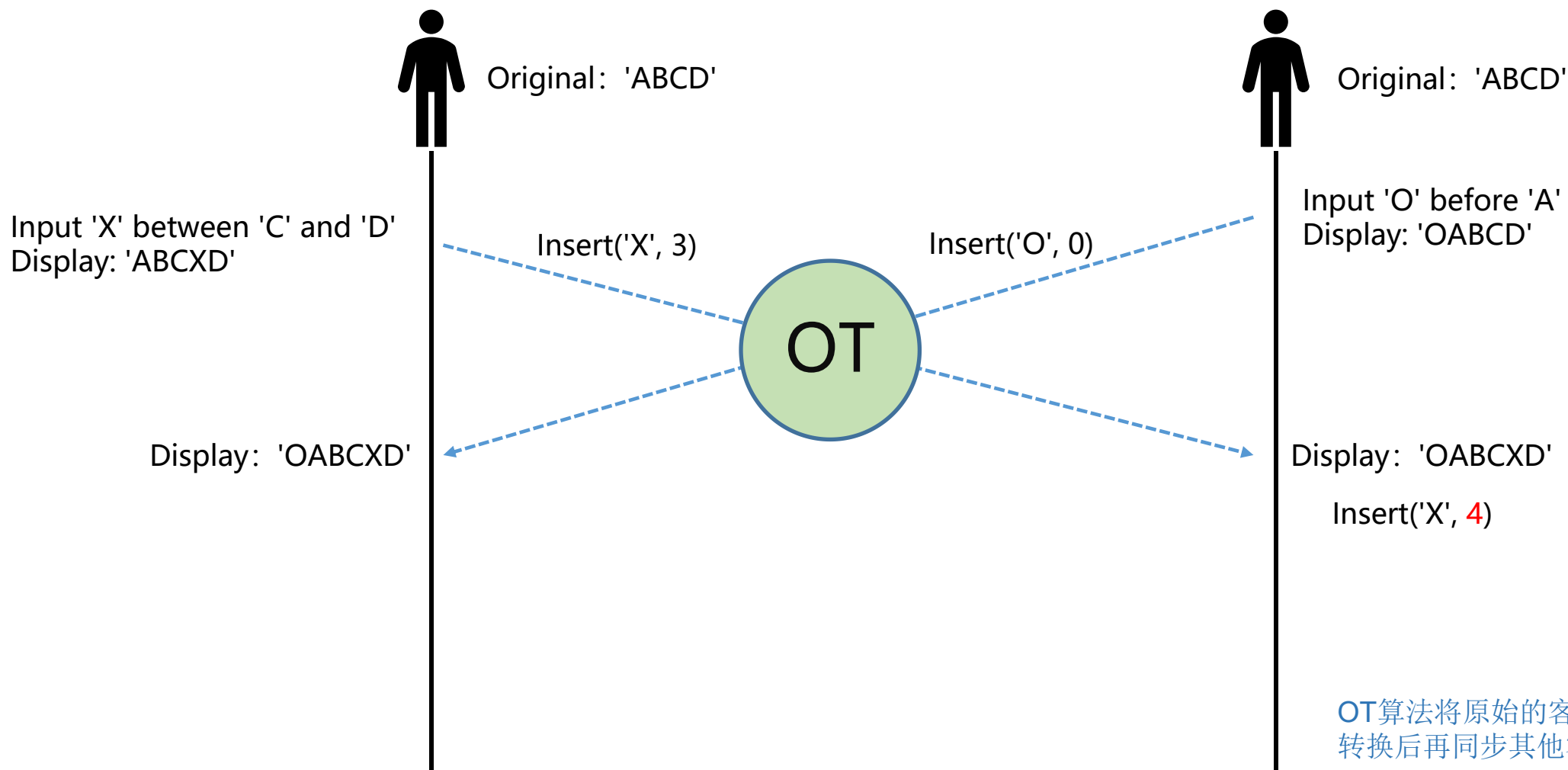
- CRDT(Conflict-Free Replicated Data Type)







# OT(Operational Transform) 算法



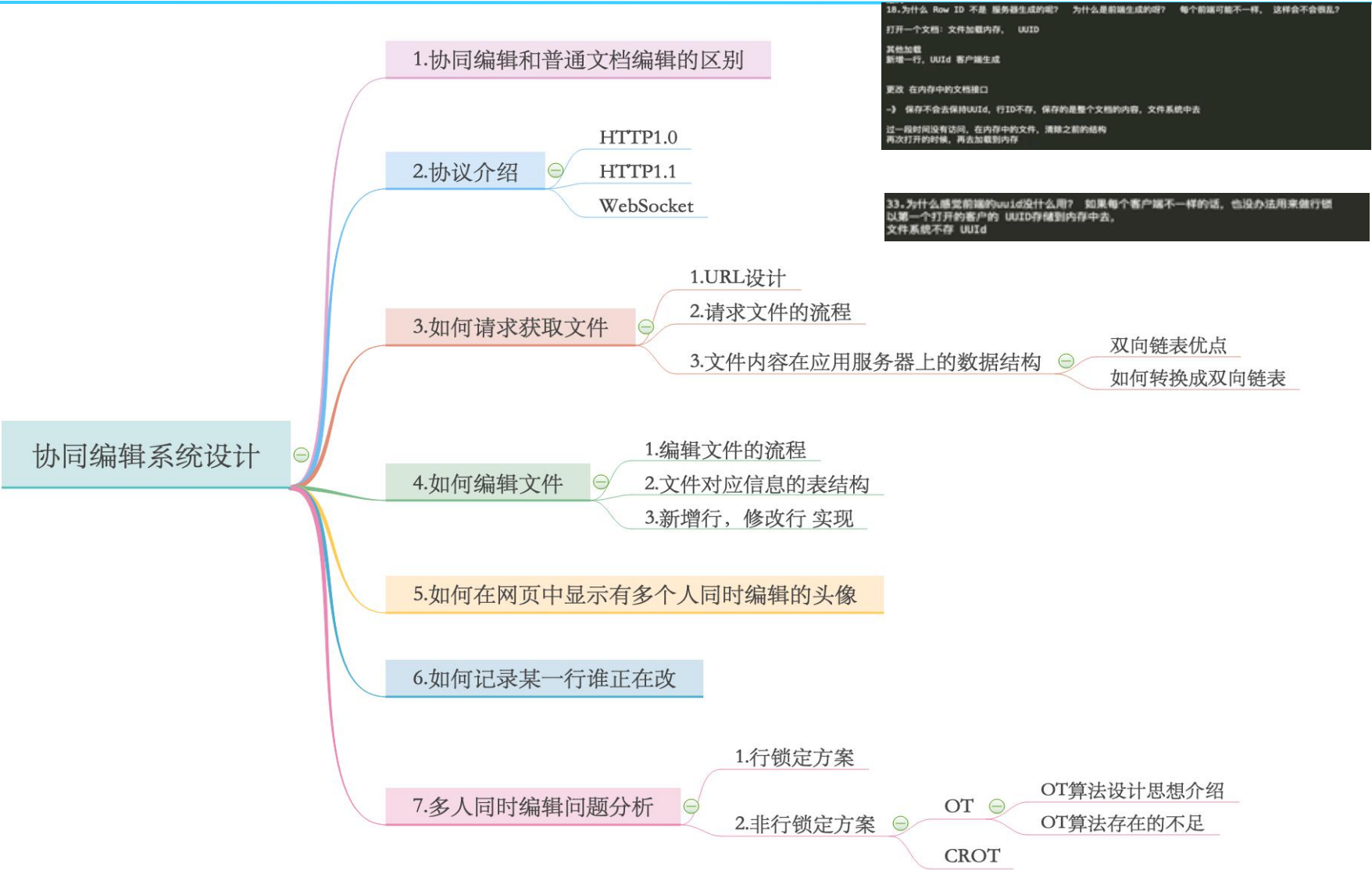
OT算法将原始的客户端指令进行了转换，  
转换后再同步其他客户端

## OT算法存在的问题

OT算法虽然解决了位置的问题，但是不能解决语义的问题，可能转换后的语义不是两个人同时想要的，不能够达到100%完美的效果；  
只能够让编辑不产生冲突，但是不产生冲突后不能保证，因为在多个人编辑是，进过OT算法的转换后语句可能是不通顺的，还需要人为的去调整。

考虑到用户体验，限制同时编辑人数

- <https://svn.apache.org/repos/asf/incubator/wave/whitepapers/operational-transform/operational-transform.html>  
G-Suite协同引擎的协议白皮书
- <https://dl.acm.org/doi/10.1145/274444.274447>  
GOT算法及一维数据操作变换算法论文
- <https://www.tiny.cloud/blog/real-time-collaboration-ot-vs-crdt/>  
To OT or CRDT, that is the question
- <https://github.com/share/sharedb>  
real-time database backend based on OT of JSON documents



# Thanks