NIO解读和应用

鲫鱼哥

CHAINUP

以下是今天讨论重点

引出话题

BIO演示

NIO演示

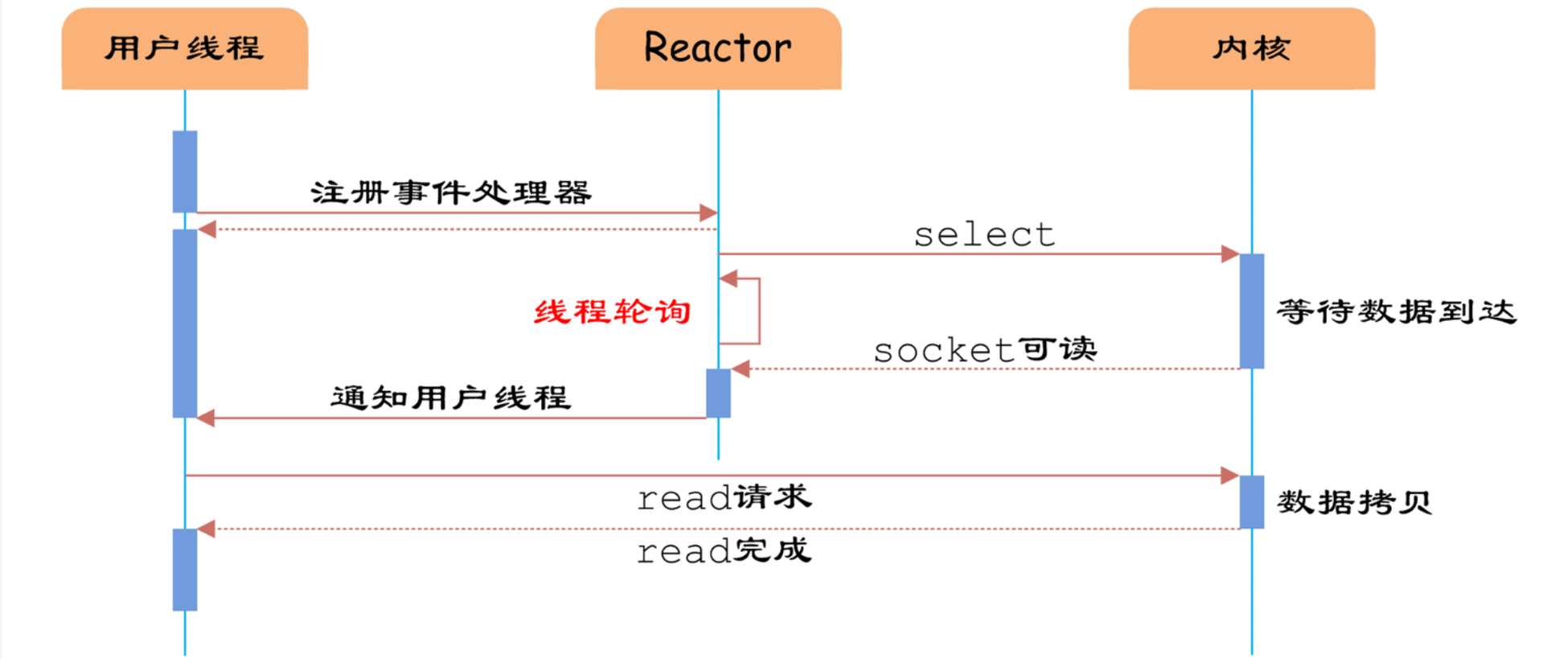
NIO架构介绍

redis中的多路复用

java中的NIO场景

CHAINUP

引出话题:任威分享redis



CHAINUP

引出话题:来个段子

老张爱喝茶，废话不说，煮开水。
出场人物:老张，水壶两把(普通水壶，简称水壶;会响的水壶，简称响水壶)。

1 老张把水壶放到火上，立等水开。(同步阻塞)
老张觉得自己有点傻
2 老张把水壶放到火上，去客厅看电视，时不时去厨房看看水开没有。(同步非阻塞) 老张还是觉得自己有点傻，于是变高端了，买了把会响笛的那种水壶。水开之后，能大声发出嘀~~~~的噪音。
3 老张把响水壶放到火上，立等水开。(异步阻塞)
老张觉得这样傻等意义不大
4 老张把响水壶放到火上，去客厅看电视，水壶响之前不再去看它了，响了再去拿壶。(异步非阻塞) 老张觉得自己聪明了。

所谓同步异步，只是对于水壶而言。
普通水壶，同步;响水壶，异步。 虽然都能干活，但响水壶可以在自己完工之后，提示老张水开了。这是普通水壶所不能及的。 同步只能让调用者去轮询自己(情况2中)，造成老张效率的低下。
所谓阻塞非阻塞，仅仅对于老张而言。
立等的老张，阻塞;看电视的老张，非阻塞。

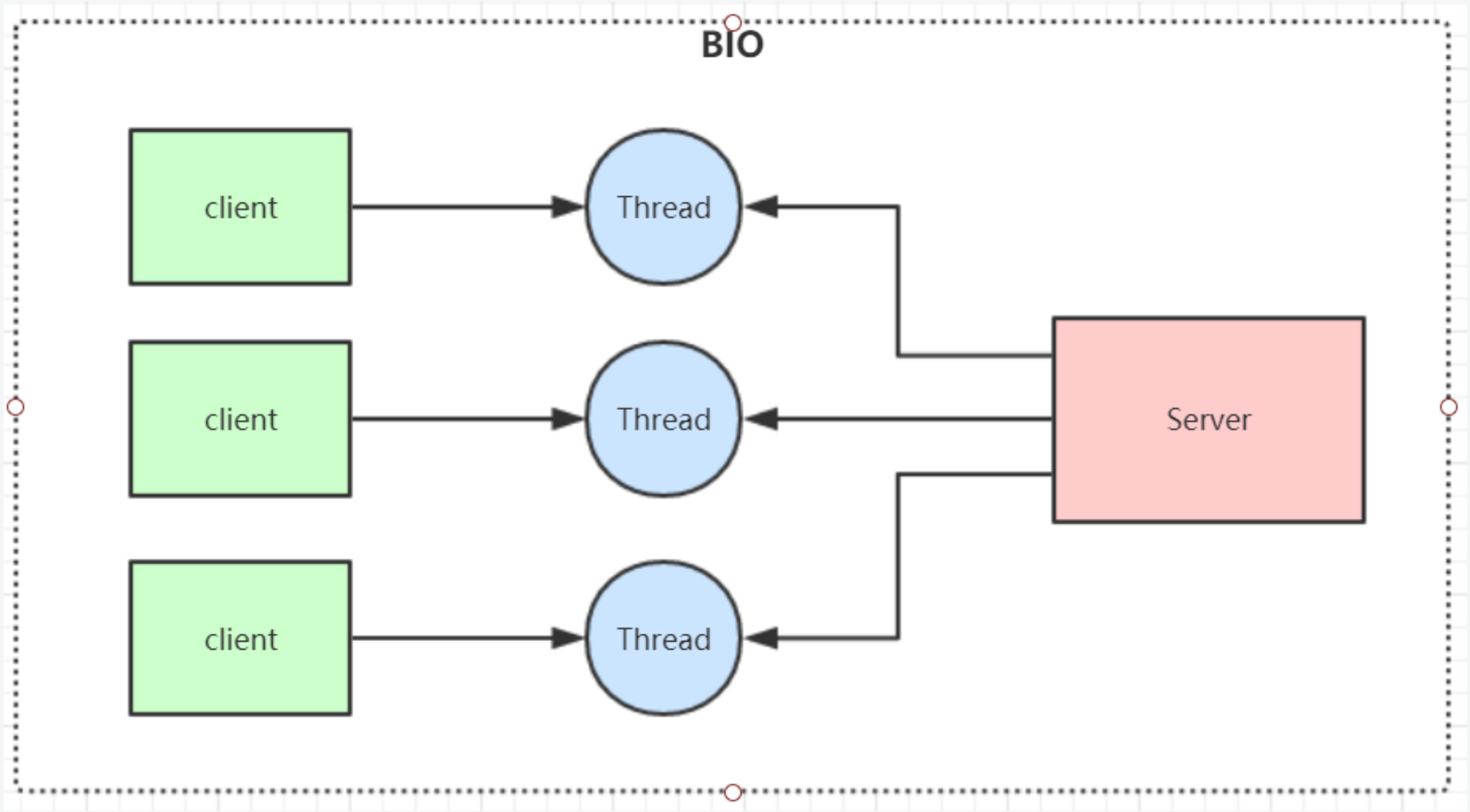
CHAINUP

BIO：阻塞IO

BIO代码演示

同步阻塞模型，一个客户端连接对应一个处理线程

C10K问题?



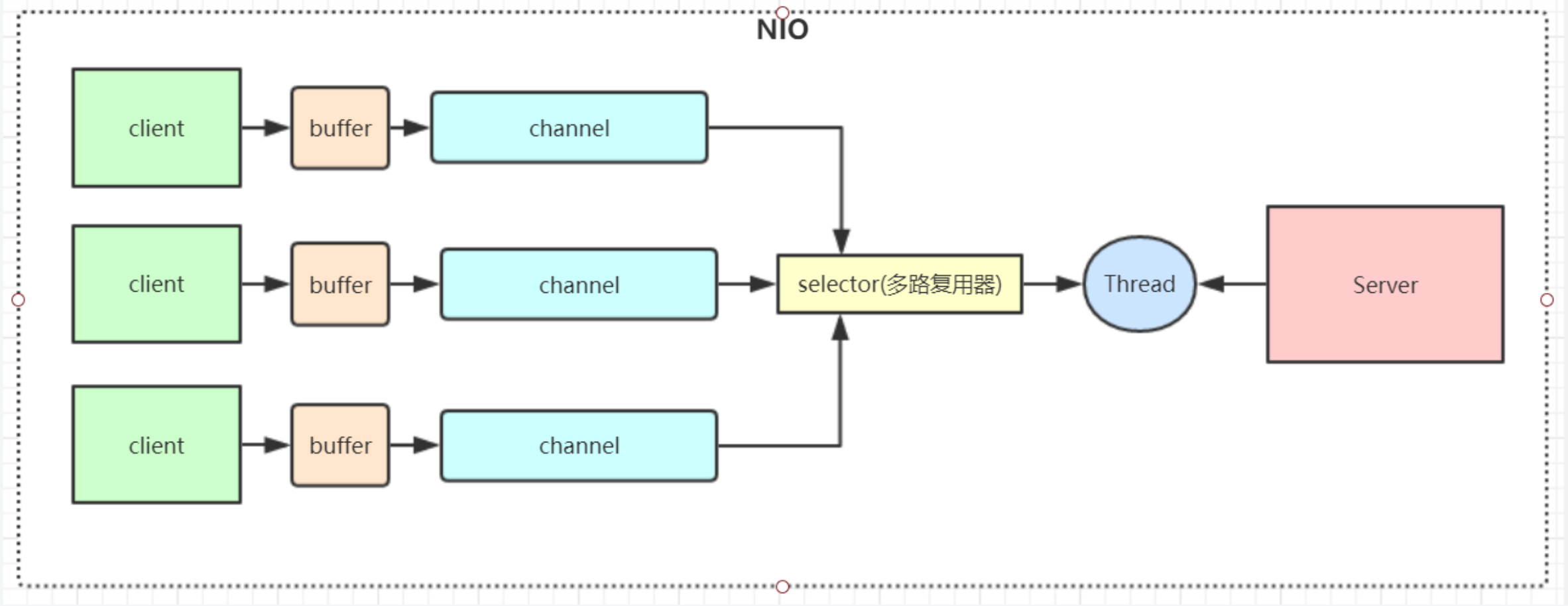
CHAINUP

NIO:同步非阻塞IO

NIO代码演示

对比BIO

IO	NIO
面向流	面向缓冲
阻塞IO	非阻塞IO
无	选择器



CHAINUP

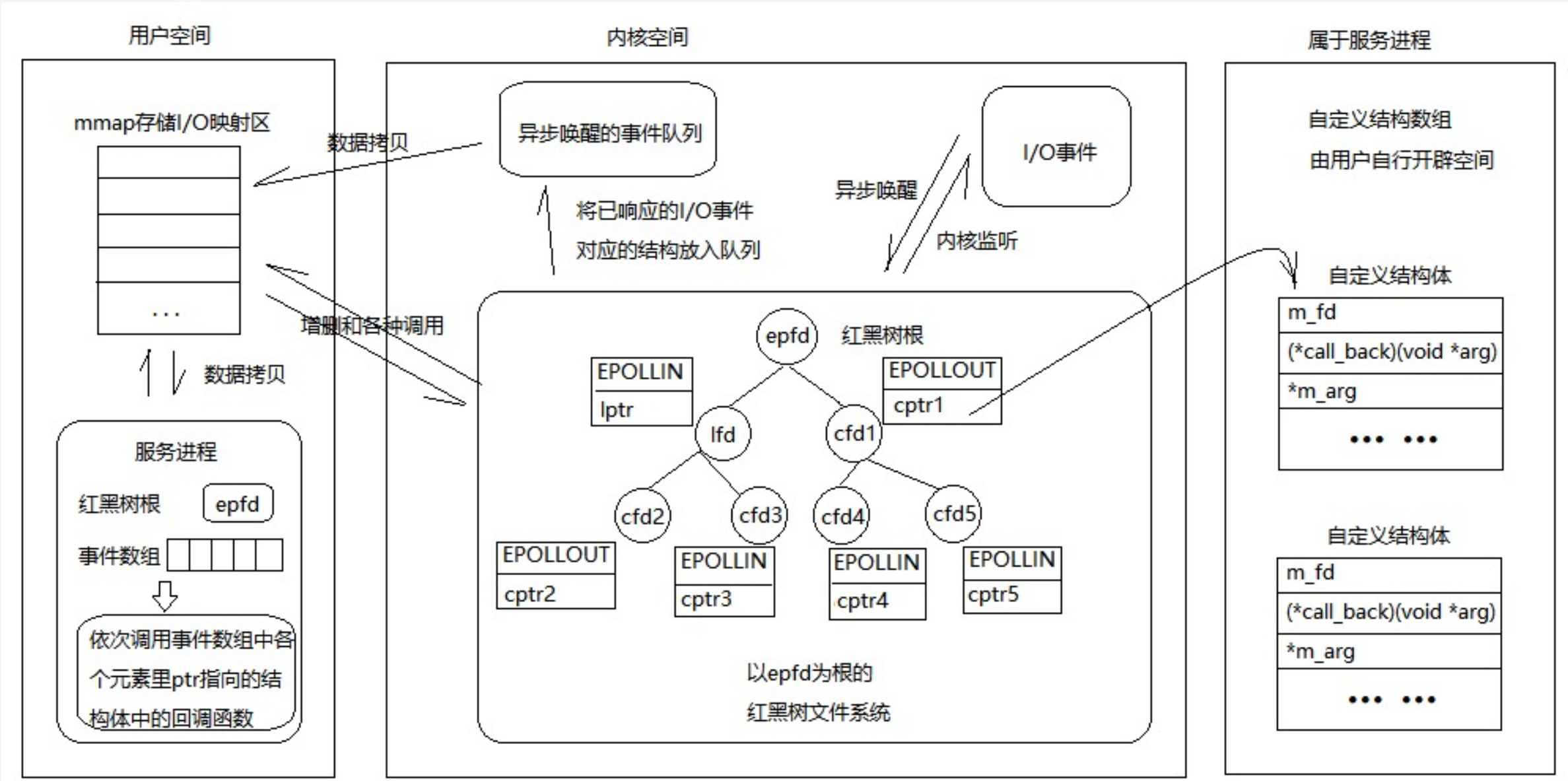
多路复用LINUX实现

具体怎么玩的可以结合redis看

	select	poll	epoll(jdk 1.5及以上)
操作方式	遍历	遍历	回调
底层实现	数组	链表	哈希表
IO效率	每次调用都进行线性遍历，时间复杂度为O(n)	每次调用都进行线性遍历，时间复杂度为O(n)	事件通知方式，每当有IO事件就绪，系统注册的回调函数就会被调用，时间复杂度O(1)
最大连接	有上限	无上限	无上限

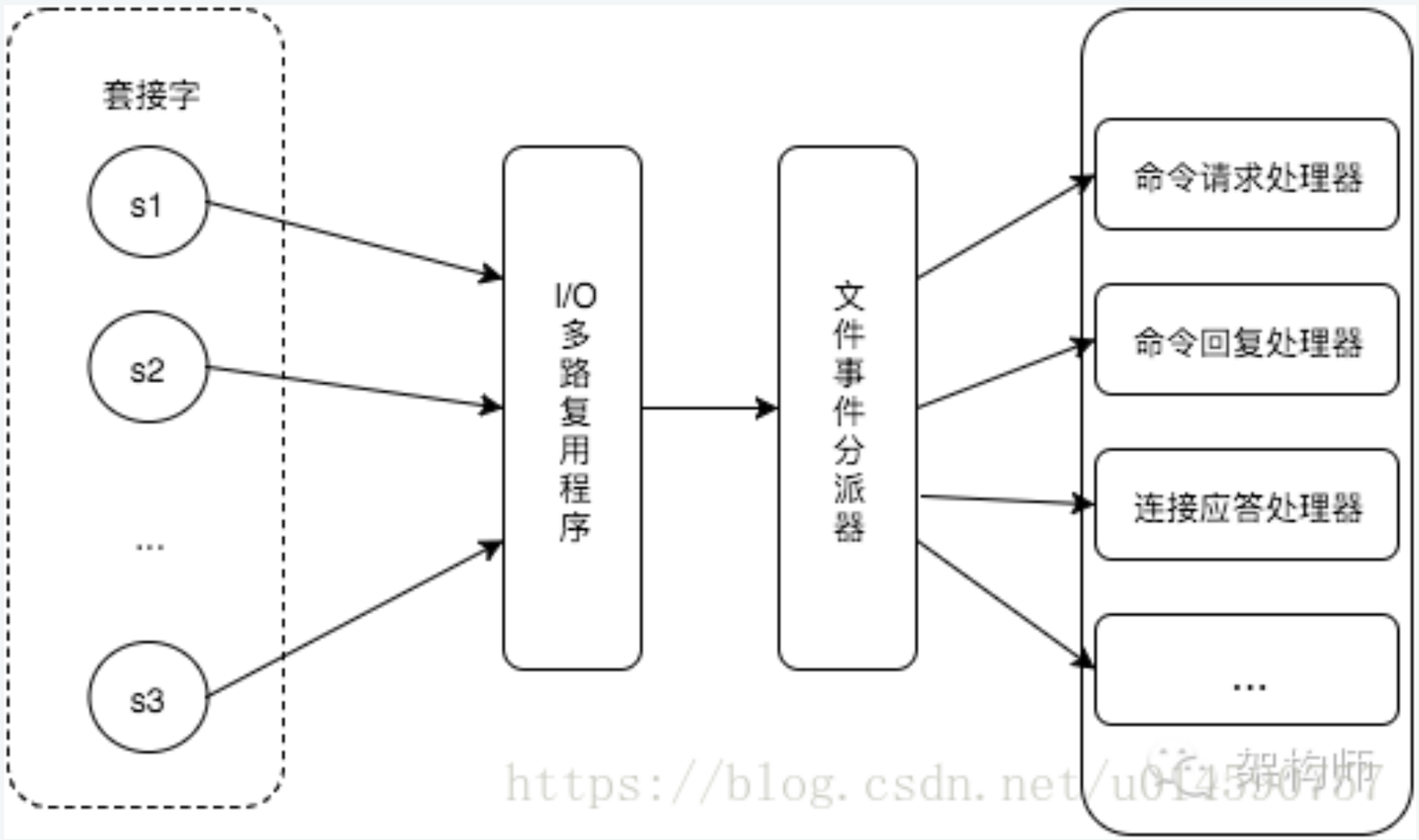
CHAINUP

图解epoll



CHAINUP

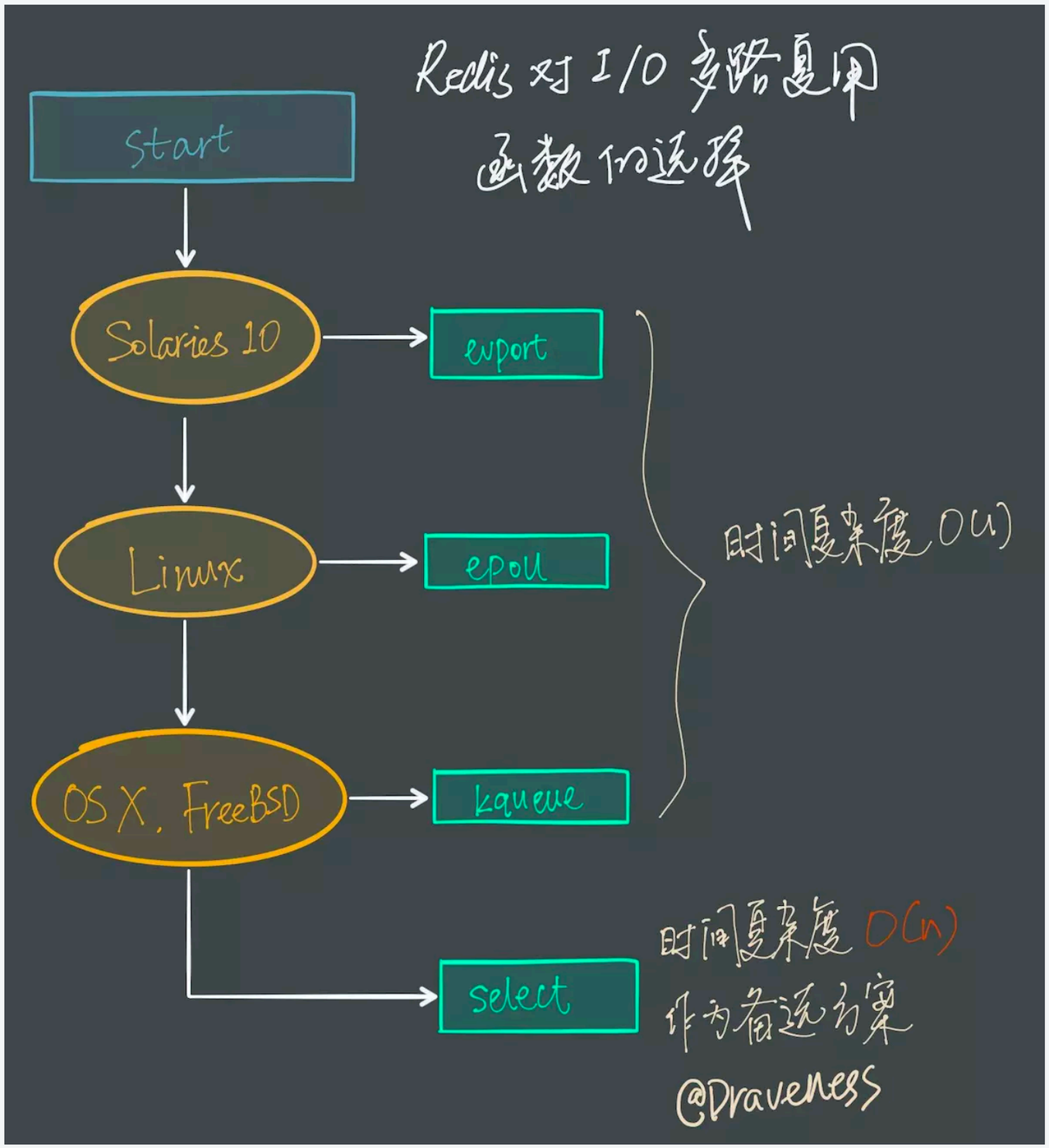
回看redis



CHAINUP

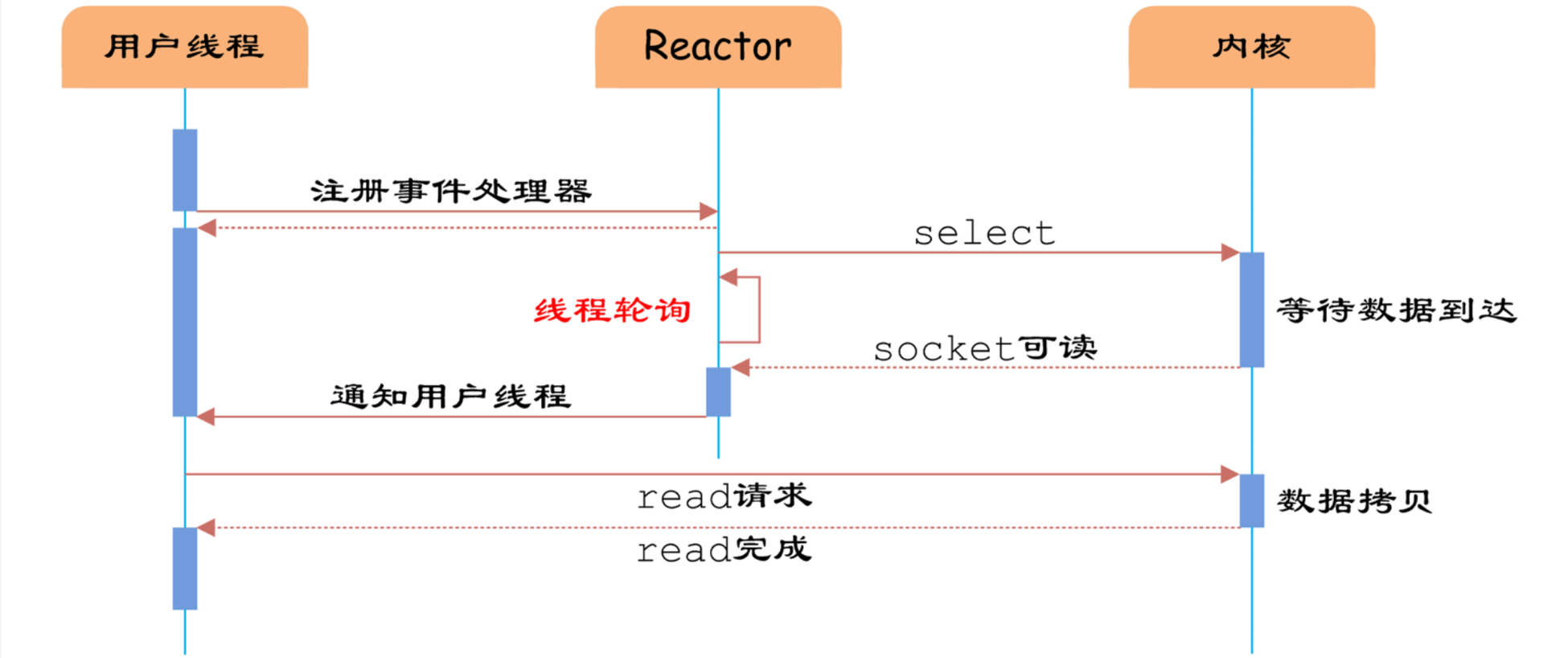
回看redis

不同操作系统
不同多路复用



CHAINUP

回看redis



CHAINUP

NIO应用

Tomcat并发优化参数

connector

acceptCount：达到最大连接数时启用的连接队列，默认队列长度为100

maxConnections：服务端允许接收和处理的最大连接数。默认10000

maxThreads：线程池中最多允许存在多少线程用于**处理请求**。默认200

Netty

解决了java nio使用复杂的痛点

Netty拥有高性能、 吞吐 量更高，延迟更低，减少资源消耗，最小化不必要的内存复制等优点

Hadoop的RPC框架Avro、RocketMQ、Dubbo等等，tomcat满足不了的网络通信，几乎都用Netty

CHAINUP

Q&A

THANKS