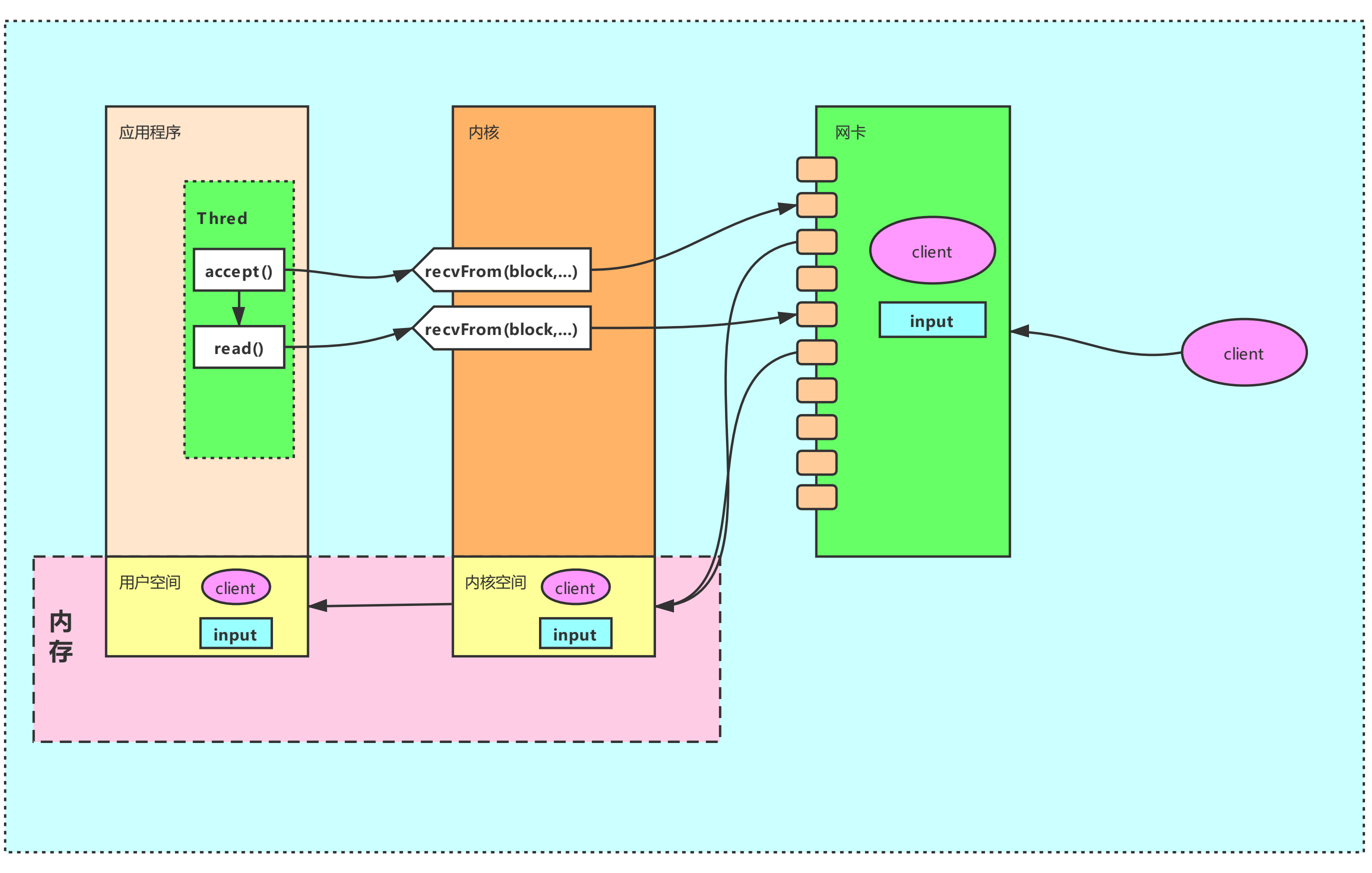
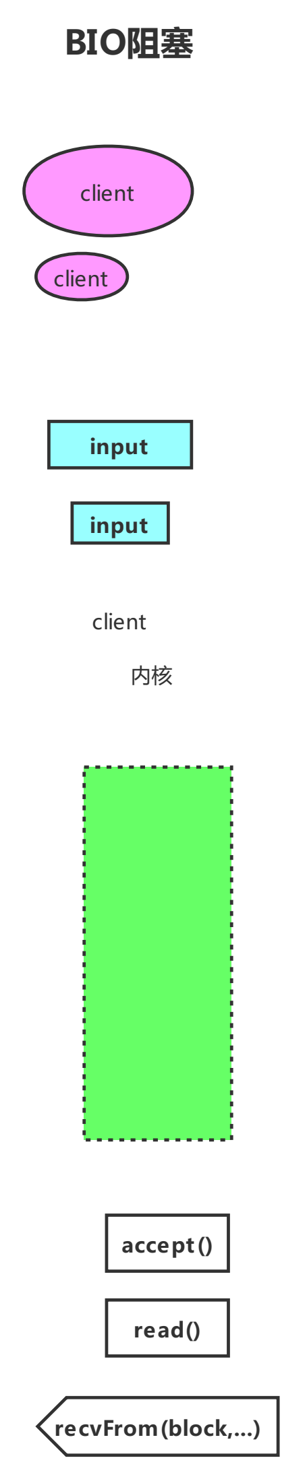
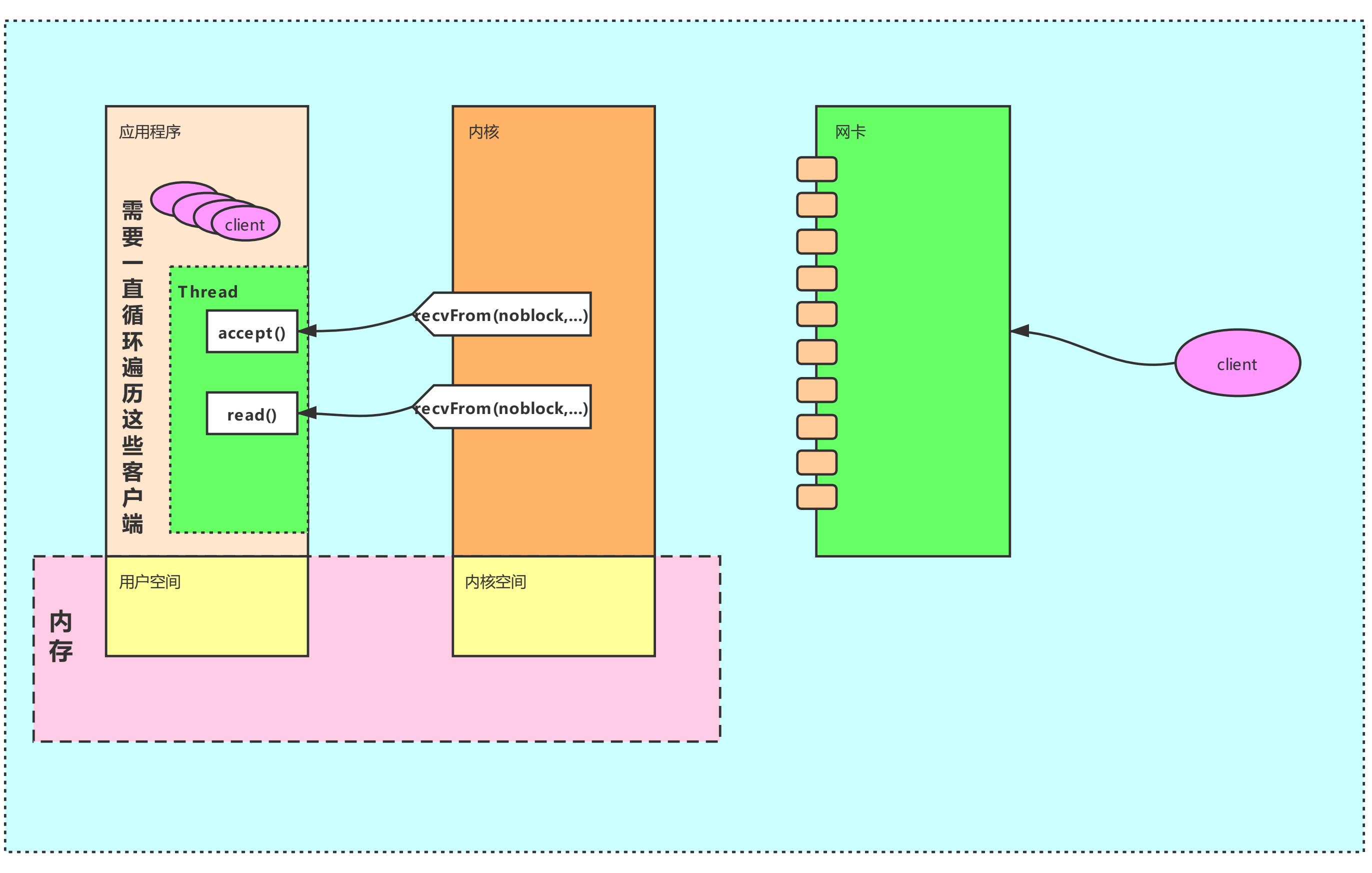
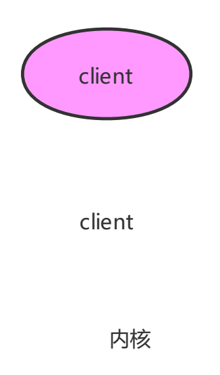
BIO阻塞





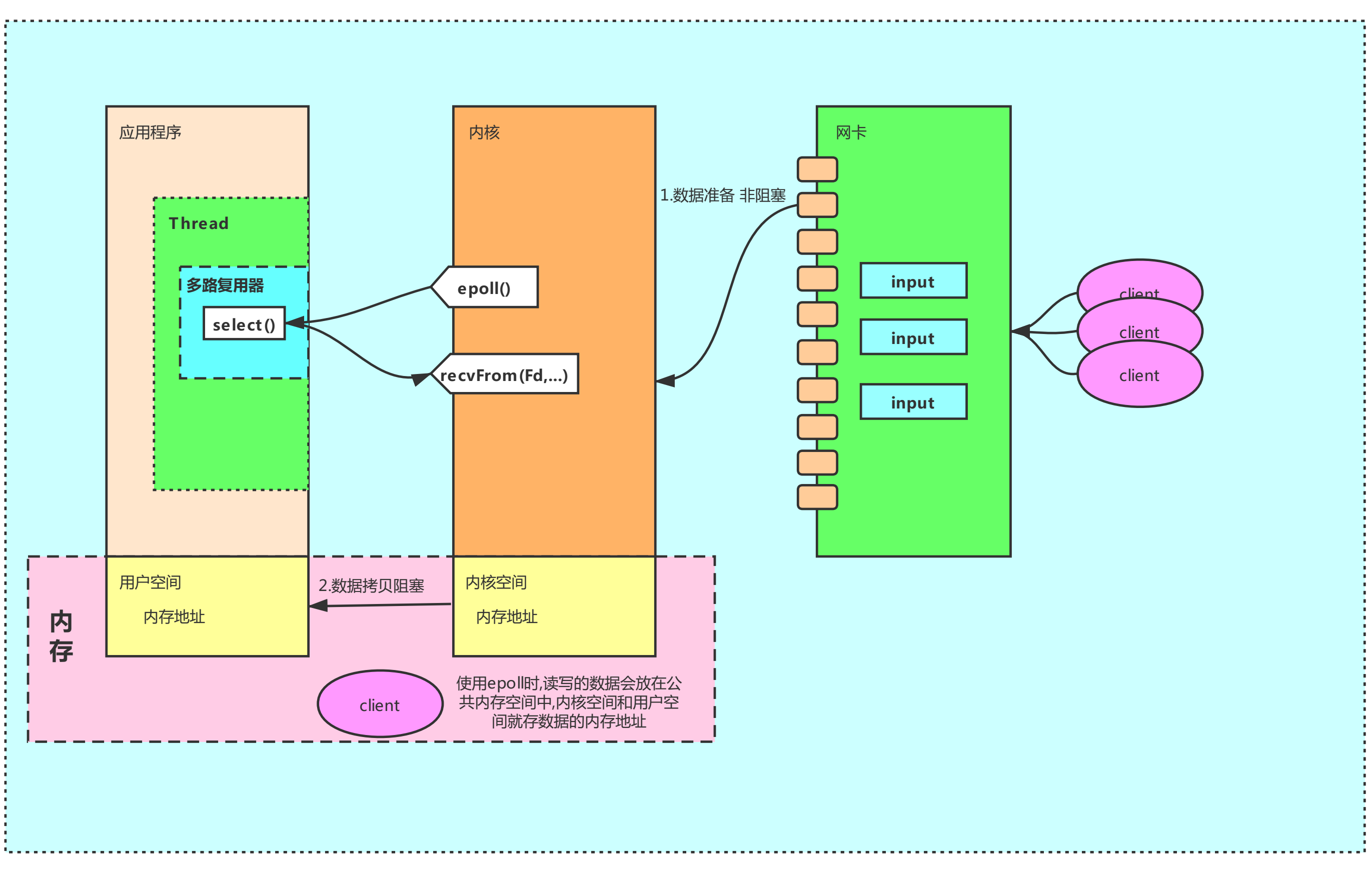
IO非阻塞



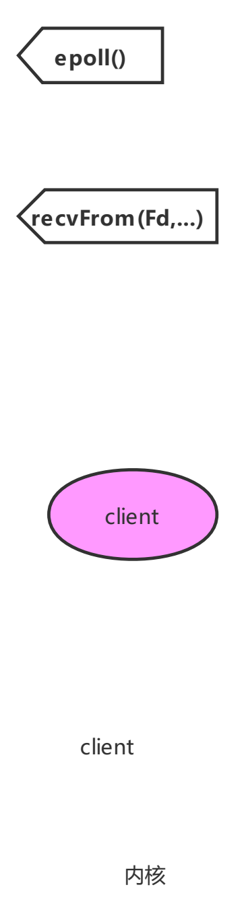


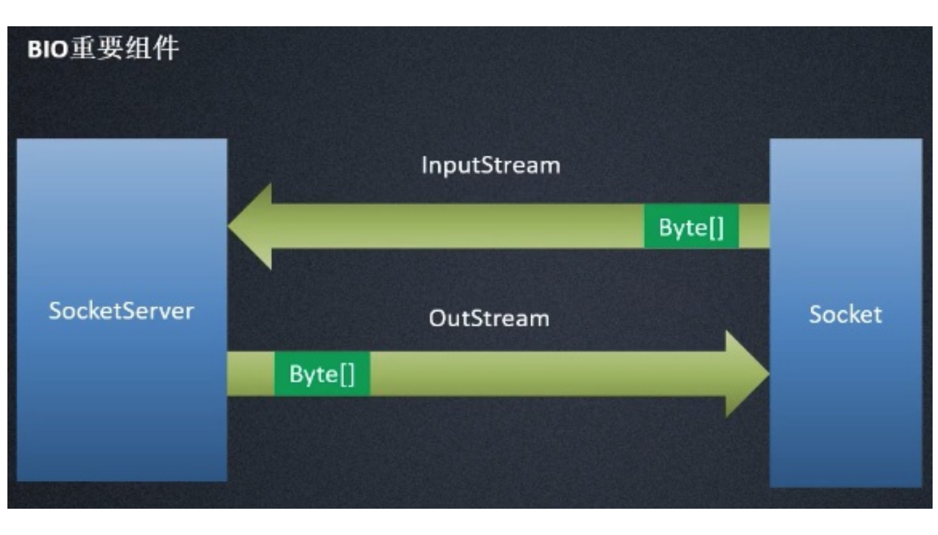
IO多路复用 非阻塞但非异步

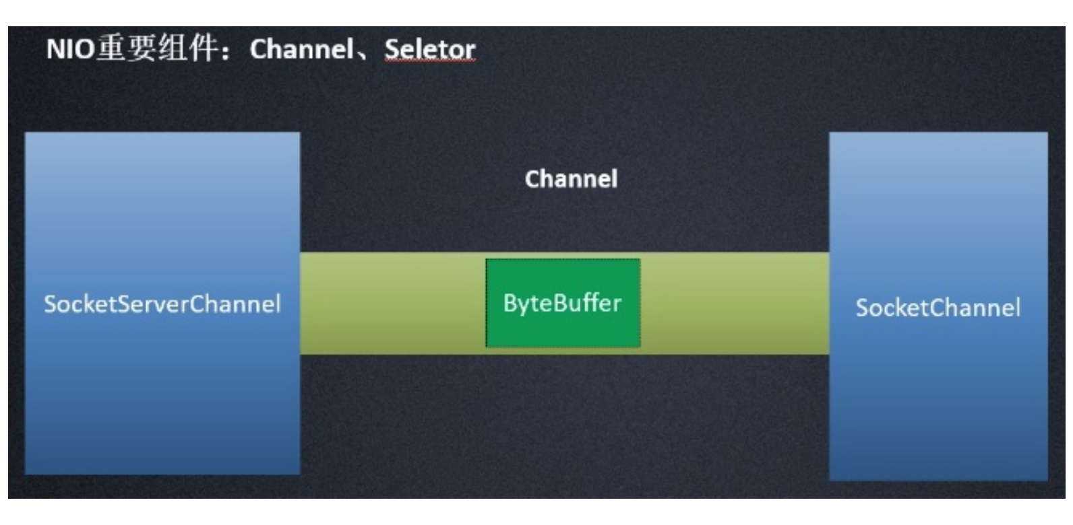
select、poll、epoll



使用epoll时,读写的数据会放在公共内存空间中,内核空间和用户空间就存数据的内存地址



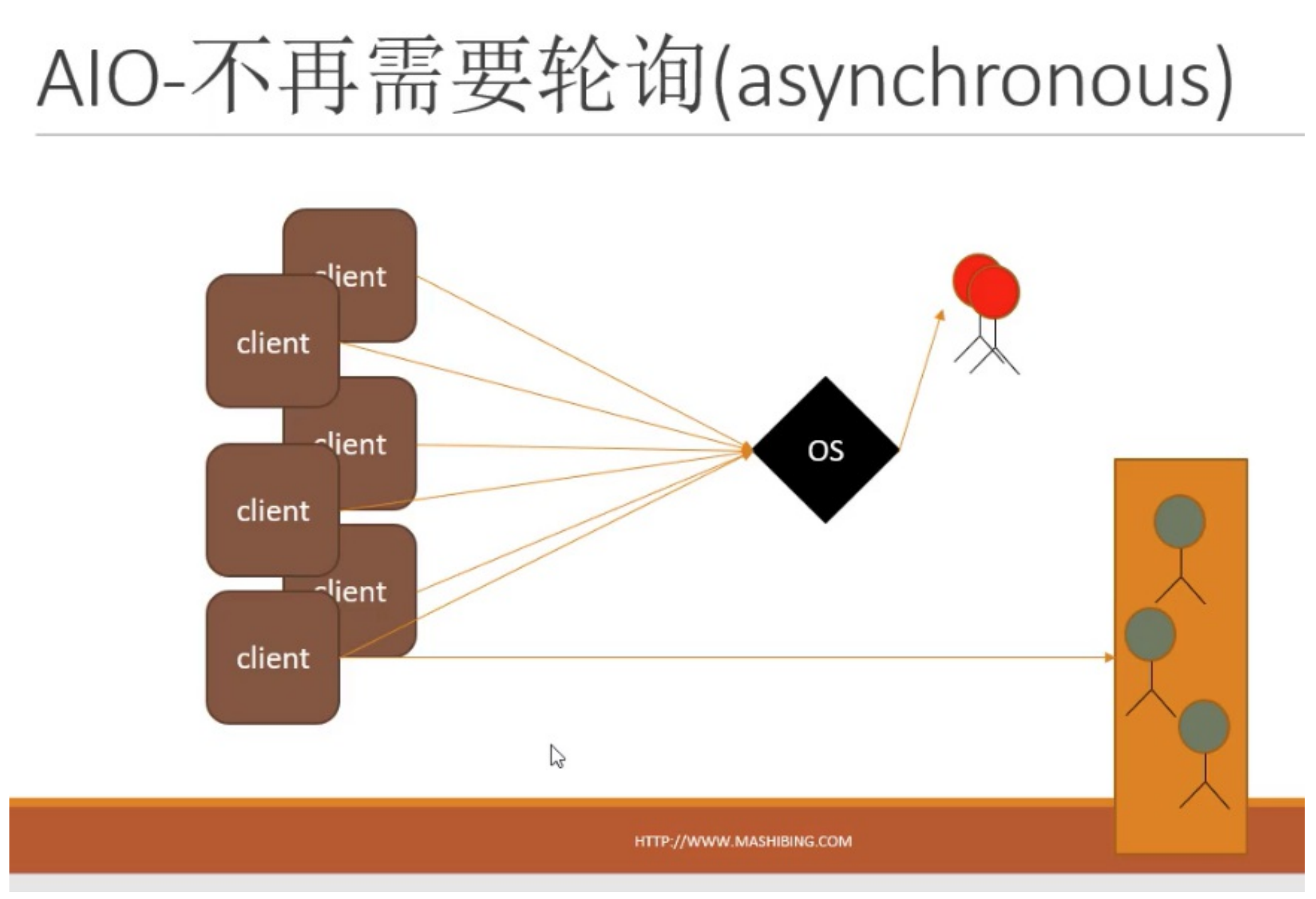




属性 ↓ I/O 模型 →	同步阻塞 I/O（BIO）	伪异步 I/O	非阻塞 I/O（NIO）	异步 I/O（AIO）
Client 数:I/O 线程	1:1	M:N(M>=N)	M:1	M:0
I/O（阻塞）类型	阻塞 I/O	阻塞 I/O	非阻塞 I/O	非阻塞 I/O
I/O（同步）类型	同步 I/O	同步 I/O	同步 I/O（多路复用）	异步 I/O
API 使用难度	简单	简单	复杂	一般
调试难度	简单	简单	复杂	复杂
可靠性	非常差	差	高	高
吞吐量	低	中	高	高

适用场景 （1）BIO方式适用于连接数目比较小且固定的架构，这种方式对服务器资源要求比较高，并发局限于应用中，JDK1.4以前的唯一选择，但程序直观简单易理解。
（2）NIO方式适用于连接数目多且连接比较短（轻操作）的架构，比如聊天服务器，并发局限于应用中，编程比较复杂，JDK1.4开始支持。

（3）AIO方式使用于连接数目多且连接比较长（重操作）的架构，比如相册服务器，充分调用OS参与并发操作，编程比较复杂，JDK7开始支持。
另外，I/O属于底层操作，需要操作系统支持，并发也需要操作系统的支持，所以性能方面不同操作系统差异会比较明显。


AIO-不再需要轮询(asynchronous)