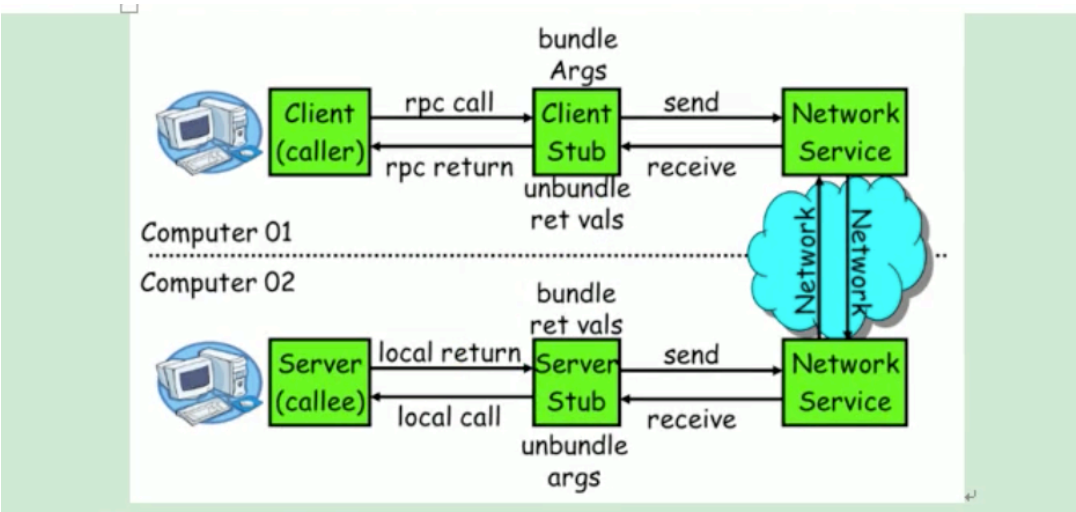
rpc的一个原理：



一次完整的rpc调用（同步调用，异步另说）如下：

（1）服务消费方(client)调用以本地调用方式调用服务

（2）client stub接收到调用后负责将方法、参数组装成能够进行网络传输的消息体；

（3）client stub找到服务地址，并将消息发送到服务端；

（4）server stub收到消息后进行解码；

（5）server stub根据解码结果调用本地的服务；

（6）本地服务执行并将结果返回给server stub；

（7）server stub将返回结果打包消息并通过网络传输发送至消费方；

（8）client stub接收到消息，进行解码；

（9）服务消费方得到最终结果。

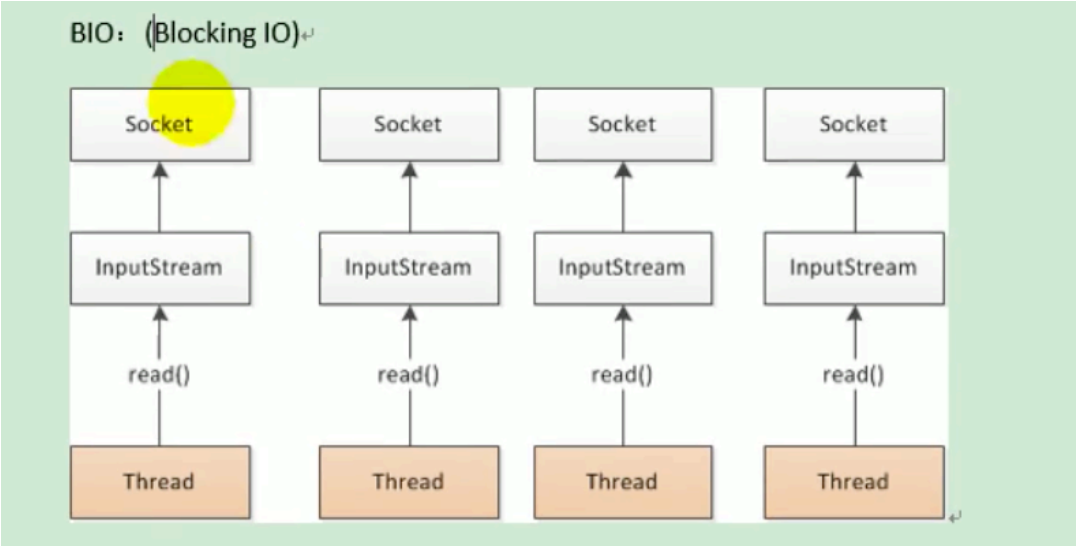
RPC框架的目的就是要把2-8这些步骤都封装起来，这些细节对于用户来说都是透明的，不可见的。

Netty

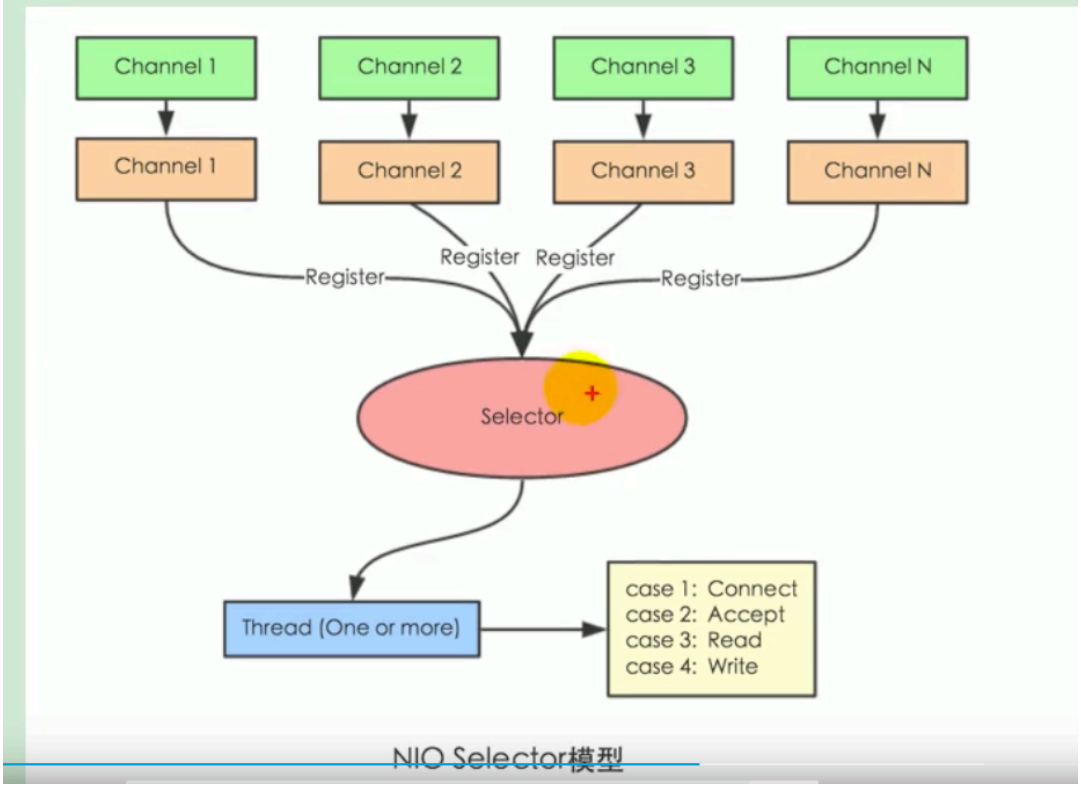
Netty是一个**异步事件驱动**的网络应用程序框架，用于快速开发可维护的高性能协议服务器和客户端。它极大的简化了**TCP和UDP套接字服务器**等网络编程。

Netty是基于NIO的。（No Blocking IO）

BIO:Blocking IO



NIO: No Blocking IO



这里多了一些概念，NIO是 selector概念：

Selector一般称为选择器，也可以翻译为多路翻译器。

通过监听Channel的几种事件，每种事件发生之后，再新建一个线程去执行。这里事件类型有：

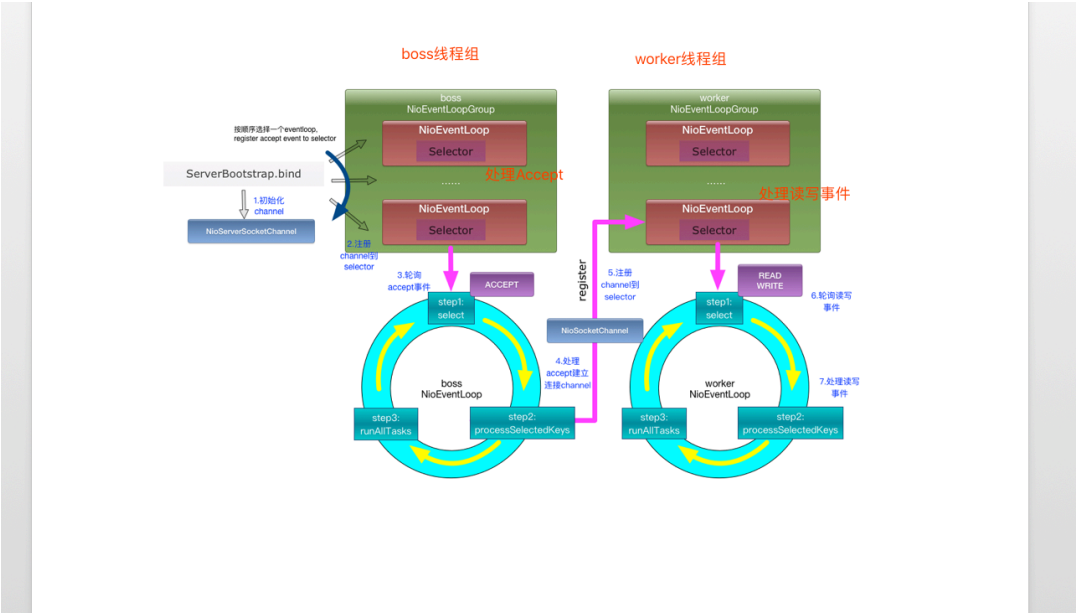
（1）Connect（连接就绪）

（2）Accept（接受就绪）

（3）Read（读就绪）

（4）Write（写就绪）

Netty的一个基本原理：



（1）serverBootstrap.bind启动时，会初始化channel NioServerSocketChannel并且注册到Selector

（2）有两个线程组： boss线程组和worker线程组，其中boss线程组中的selector是监听的accept事件，而worker中的selector是用来监听读写事件的，之后轮询读写事件并且维护一个队列去处理这个读写事件。