1.同步和异步

首先先明确几个概念

I/O模型中的同步以及异步

1. 同步：所谓同步，就是在发出一个功能调用时，在没有得到结果之前，该调用就不返回。**也就是必须一件一件事做**,等前一件做完了才能做下一件事。

2. 异步：异步的概念和同步相对。当一个异步过程调用发出后，调用者不能立刻得到结果。实际处理这个调用的部件在完成后，通过状态、通知和回调来通知调用者。（不需要知道该功能结果，该功能有结果后通知我（回调通知））

3. 阻塞

4. 非阻塞

五种IO模型

1. 阻塞I/O
2. 非阻塞I/O
3. I/O复用（select/poll）
4. 信号驱动I/O
5. 异步I/O

并发模型中的同步以及异步

1. 同步：完全按照代码的执行顺序执行
2. 异步：程序的执行需要由系统事件来驱动（中断定时器信号等）

半同步半异步模型

1. 同步线程：按照并发模型中同步方式运行的线程
2. 异步线程：按照并发模型中异步方式运行的线程
3. 半同步半异步模型：同时使用同步线程以及异步线程

- 在这个模型中同步线程用来处理客户逻辑，异步线程用来处理I/O事件

领导者追随者模式

1. 领导者追随者模式是多个工作线程轮流获得事件源的资源，轮流监听、分发并处理事件的一种模式。
2. 在任意时间点，程序仅有一个领导者线程，它负责监听I/O事件。而其他线程则都是追随者，他们休眠在线程池中等待成为新的领导者。当前的领导者如果监测到I/O事件，首先要从线程池中推选出新的领导者线程，然后去处理I/O事件。从而实现了并发
3. 领导者追随者包含如下几个组件：句柄集（Handleset）、线程集（ThreadSet）、事件处理器（EventHandler）和具体的事件处理器

- 句柄集：通常是一个文件描述符的集合：监听句柄上的I/O事件，并将集中的就绪事件通知给领导者线程。领导者则调用绑定在Handle上的事件处理器来处理事件

```
1
```

- 线程集：这个组件是所有工作线程的管理者。它负责各线程之间的同步，以及新领导者的推选。线程集中的线程在任意时间肯定处于如下三种状态之一

leader:线程当前处于领导者身份，负责等待句柄集上的I/O事件

Processing：线程正在处理事件。领导者检测到I/O事件之后，可以转移到Processing状态来处理事件，并调用Promote_new_leader方法推选新的领导者；也可以指定其他追随者来处理事件，此时领导者的地位不变，当处于Processing 状态的线程处理完事件之后，如果当前线程集中没有领导者，则它将成为新的领导者，否则直接转变为追随者

Follower:线程当前处于追随者身份，通过调用线程集的join方法等待成为新的领导者，也可能被当前的临到者指定来处理新的任务

- 事件处理器和具体的事件处理器

事件处理器通常包含一个或者多个回调函数handle_event，这些回调函数用于处理事件对应的业务逻辑。事件处理器在使用前需要被绑定到莫格句柄上，当该句柄上有事件发生时，领导者就执行与之绑定的事件处理器中的回调函数

两种高效的事件处理模式

随者网络设计模式的兴起Reactor模式和Proactor事件处理模式应运而生

Reactor模式：要求主线程（I/O处理单元）只负责监听文件描述符上是否有事件发生，有的话立即将该事件通知工作线程（逻辑单元）。除此之外，主线程不做任何其他实质性的工作。读写数据，接受新的连接，以及处理客户请求均在工作线程中完成

Proactor模式：Proctor模式将所有的I/O操作都交给主线程和内核处理。工作线程仅仅负责业务逻辑

有限状态机

前面的是服务器的I/O处理单元、请求队列和逻辑单元之间协调完成任务的各种模式，有限状态机是逻辑单元内部的一种高效的编程方式。

通俗易懂：

```
https://zhuanlan.zhihu.com/p/47434856
```