连接关闭流程

关闭连接分强制关闭跟优雅关闭，普通关闭，强制关闭

其中优雅关闭只关闭本端，对端关闭由对方决定，一般在内网服务器之间使用这个方式，在关闭本端前，会保证已经在发送缓存中的数据被发送。
普通关闭其他跟优雅关闭一样，但是对端关闭受本端影响，在发送普通关闭后，如果发送缓存中没有数据，或者发送缓存中的数据在接下来一段时间内被发送完成，就会关闭发送跟接收，而不等待对端关闭。
强制关闭直接丢弃发送缓存中的数据，关闭链路。

对于对端主动发起的关闭，recv会返回0，但是这个只会关闭接收，并且网络层不会通知逻辑层，如果逻辑层没有主动的shutdown或者试着发送数据，该链路是不会被断开的也就不会被释放。这里有一个情况要说明，如果对端关闭写之后直接结束了进程，或者操作系统崩溃，或者也关闭了读，对于select模式是没有任何响应的，但是在linux下的epoll下，隔一段时间（2分钟）会收到rst包，导致收到HUP跟ERR事件，这里着他们转换成读写事件，如果此时发送缓存中没有数据，是不会有任何反应的，连接断开回调也不会触发，需要逻辑层主动shutdown或者send（心跳）来触发真正的错误，最后关闭链路。

这里关闭链路的逻辑是 先shutdown -> 关闭写 -> 挂到释放列表中 在下一次循环的时候来查看释放列表，如果有元素，就释放。这里不是在close时直接释放链路有这么几方面考虑：
如果立刻删掉，比如在recv时删掉，但是这个循环中还有另外的事件，处理这个事件的时候ptr是野的了
在处理这个链接的过程中释放了另一个链接，在操作那个链接的时候，野掉了。
在处理这个链接的过程中释放了另一个链接，然后又有一个新的链接进来并且重用了刚才释放的内存，然后旧的链接有事件到来，然后新的链接就被旧的链接操作了。
所以这里选择统一来释放比较妥当。

关闭状态图