### DTP说明

**设计初衷（是什么）**

目的就是仿照FTP进行数据/文件的传输。由DTP进行传输格式（传输协议）的指定，方便发接收端了解接收到的数据是什么，以及怎么处理。方便用户使用。

**设计理由（为什么）**

理由就是当用户想要传输数据或者需求数据的时候，没有办法通知对端。于是就设计了一个传输协议，规定了传输过程中的报文格式。便于控制状态。

**工作流程**

DTP算是单独的一个组件，依赖Multiplexer，并不暴露给Entity。当用户调用Mul的SendFile/SendData的时候，Mul内部是直接调用DTP的类进行实现的。

**发送流程**

在DTP中进行控制报文的生成，通知对端本端要做什么事情，并等待对端回应。若对端同意接收/发送文件/数据，则进行文件/数据的发送。此时数据的发送依赖新建立的TCP/UDP/RDMA连接，而不是之前发送控制报文的连接。建立什么链接由用户在初始的时候指定。

**接收流程**

对端Mul在接收到控制报文之后，进行从初步的拆包，判断这个包是交付给DTP处理还是直接通知Entity处理。当DTP处理的时候，进行进一步拆包，判断对端想要做什么，然后通知用户，等待用户的相应（同意/不同意）。若同意，则与对端建立一条数据连接，专门用来传输此次数据，当数据传输完之后就销毁。

### 关于TCP/UDP/RDMA合并不到一起

应该说不算是合并到一起吧，问题在于TCP/UDP/RDMA的发送/接收逻辑并不完全一样，想用一套通用的代码进行数据的发送/接收比较困难。

**解决方案**

解决方案就是当用户初始化用什么发送的时候，进行保存。在代码里面用if进行判断，在一个函数里面用不同的代码实现。

**现阶段的状况**

传输File的时候只能用sendfile进行传输。传输数据（data）的时候可以进行选择。