消息中间件传输大文件的一种可能的改进策略

袁佳哲 谢健祥 徐荪睿

2022-5-30

1 ActiveMQ文件消息类型

- ByteMessage
  - ActiveMQ进行字节传输使用的消息类型（小文件）
- StreamMessage
  - ActiveMQ进行流传输使用的消息类型（小文件）
- BlobMessage
  - ActiveMQ中进行传输较大的文件所用消息类型

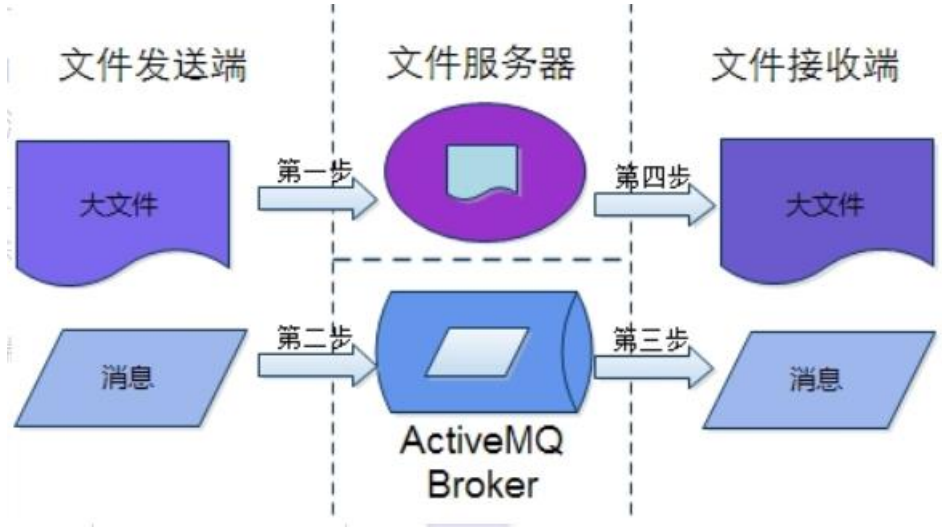
- 目前Blob方式存在的不便之处：
- 如：
  - 需自搭文件服务器（自ActiveMQ5.14后不再支持自带fileserver）
  - TCP方式传输速率低
  - 超大文件传输（如≥3GB）速度无保证

2 改进思路

➢ **源码阅读：**

ActiveMQ BlobMessage传输核心思想——利用fileserver进行文件中转

简单介绍：BlobMessage对文件中转的封装



发送端：

1. 调用Producer.send
   把文件通过PUT方式发送到FileServer
2. 将文件下载的url写入消息
3. 将消息写入broker

接收端：

1. 接收消息，发现是BlobMessage，从消息中获取url
2. 以GET方式获取文件数据
3. 以DELETE方式删除FileServer中的文件

改进核心点：

——如何更快的将文件（大文件）上传、下载fileServer

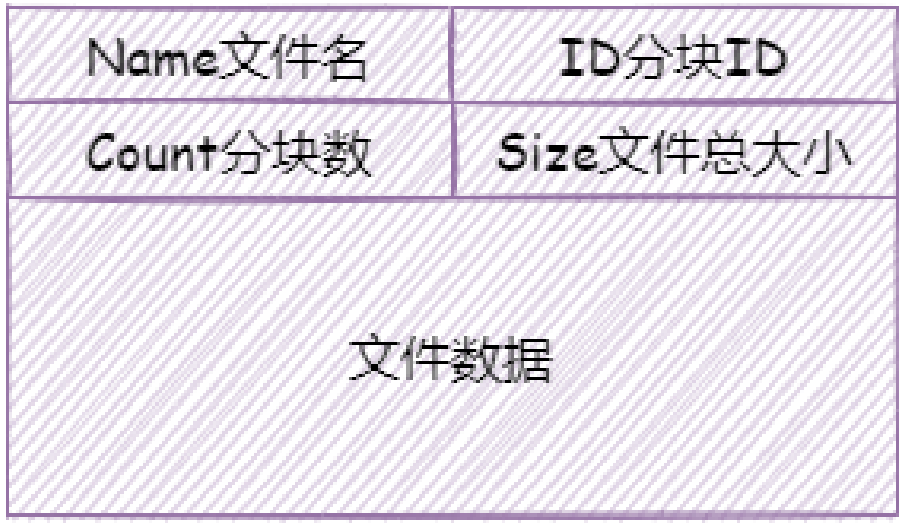
3 主要着手点

- 大文件处理：分块File Chunk
- 快速传输：UDP封包 + 可靠性保证（GBN协议）
- 稳定传输：流量控制 + 拥塞控制
- 并发传输：多线程 + 多端口

3.1 大文件分块

- 文件过大，不可能一次性把文件读取到内存当中
- 因此我们需要把文件划分成多个块，然后读取文件的指定部分，发送给接收方

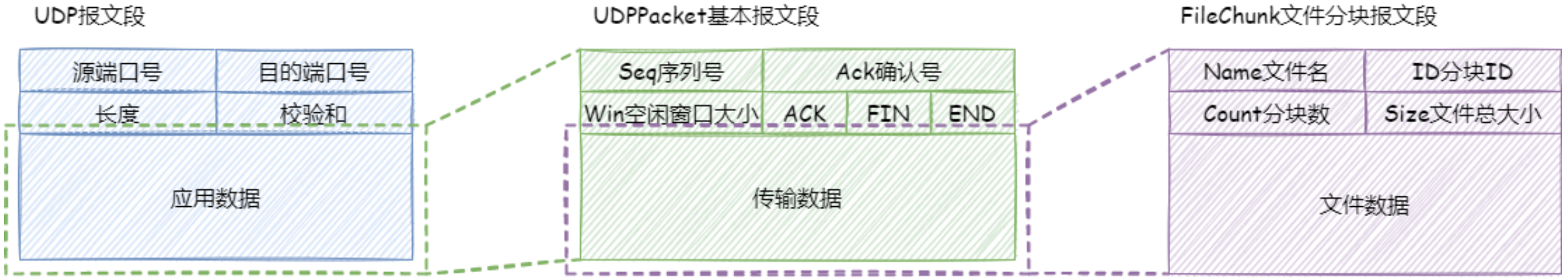
FileChunk文件分块报文段



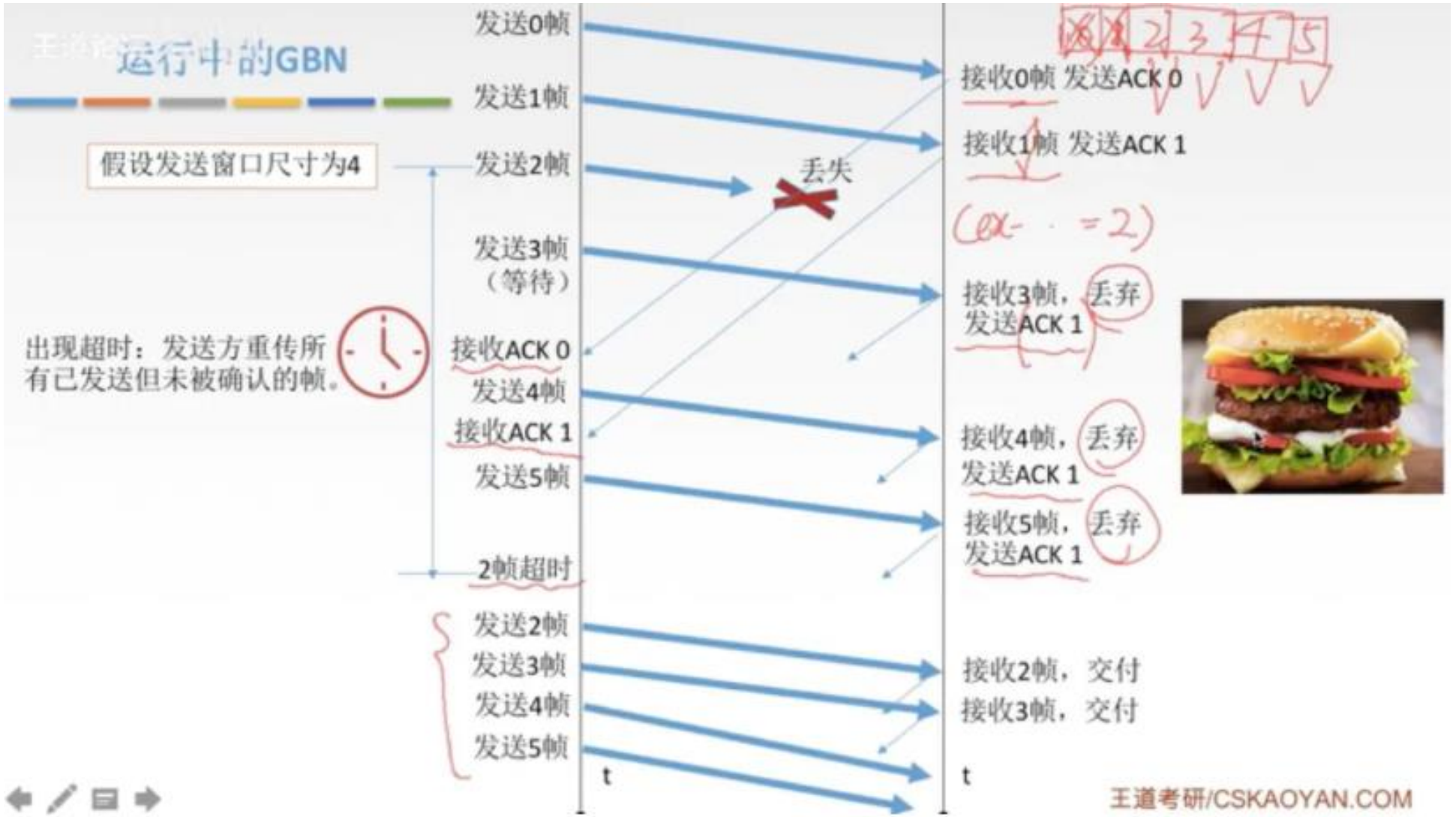
当接收方收到这个数据的时候
将指定的数据通过 **RandomAccessFile**
写入文件的指定位置

3.2 UDP封装

- 本程序基于UDP报文段，在UDP基础上封装了两层结构
- 基于Java的DatagramSocket / DatagramChannel实现



3.2 GBN（Go-Back-N）



3.3.1 流量控制

- 发送方通过封装在UDPPacket中的剩余缓存空间进行控制
  - 在接收到ACK的时候：
    - 接收缓存空闲空间小于100：减少窗口大小
    - 接收缓存空闲空间小于10：窗口大小降低为1
  - 根据接收方的缓存大小，及时调整发送方的发送速率，以控制流量大小

3.3.2 拥塞控制（参考TCP）

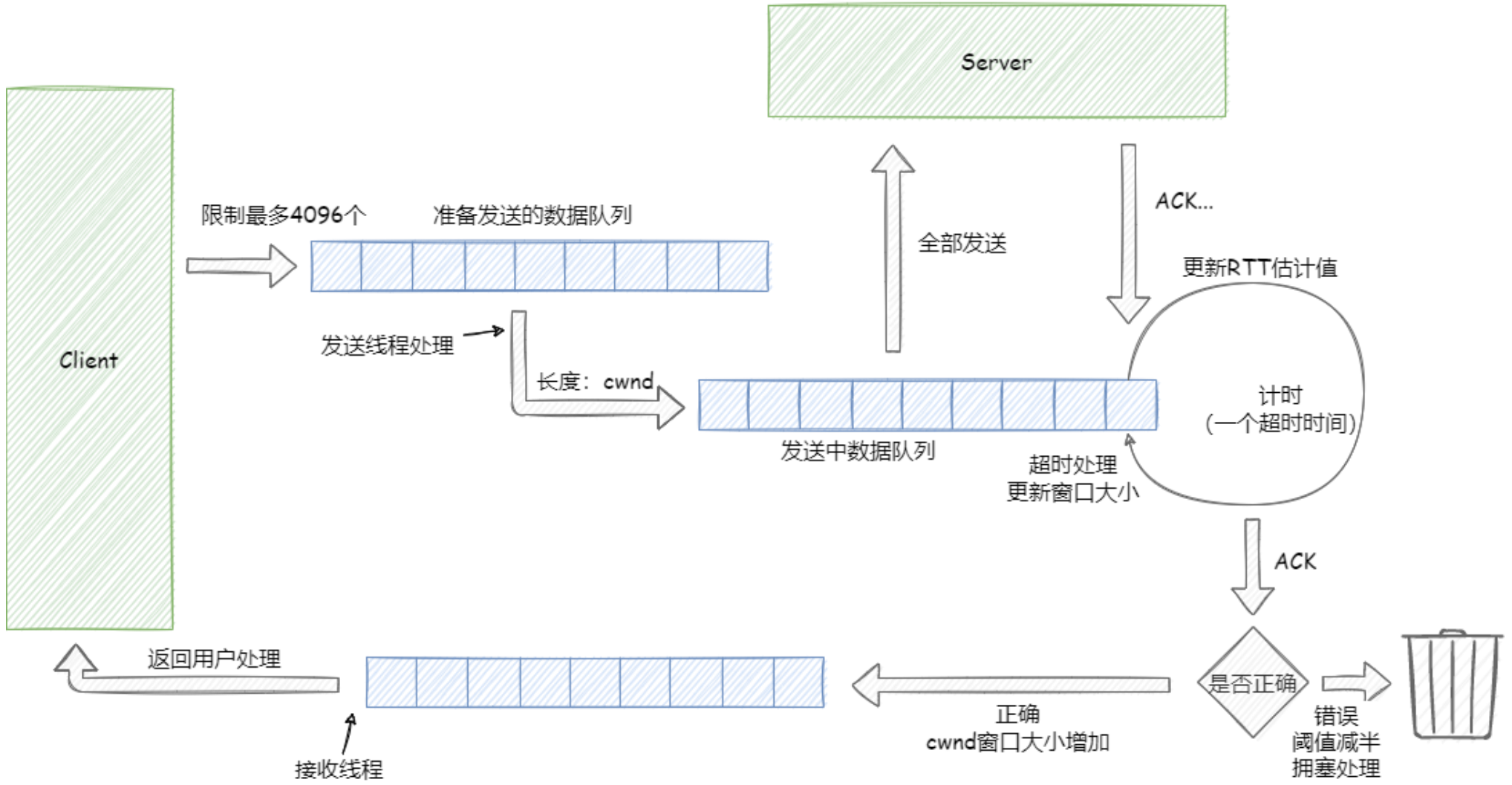
- 在接收到ACK的时候：
  - 正确的ACK，增加窗口大小：
    - cwnd <= ssthresh: cwnd = cwnd * 2
    - cwnd > ssthresh: cwnd = cwnd + 1
  - 错误的ACK，减少窗口大小
    - ssthresh变成之前的cwnd的一半
    - 重复三次以上：cwnd调成1
    - 重复三次以下：cwnd调成ssthresh + 1
  - 超时，减少窗口大小
    - ssthresh变成之前的cwnd的一半
    - cwnd调成ssthresh + 1

3.4 多端口

- 参考FTP被动模式：
  - 客户端向服务端指定端口发送请求命令（上传、下载）
  - 服务端打开新的数据传输端口，将新端口号发回客户端
  - 客户端使用新端口号发送或接收数据

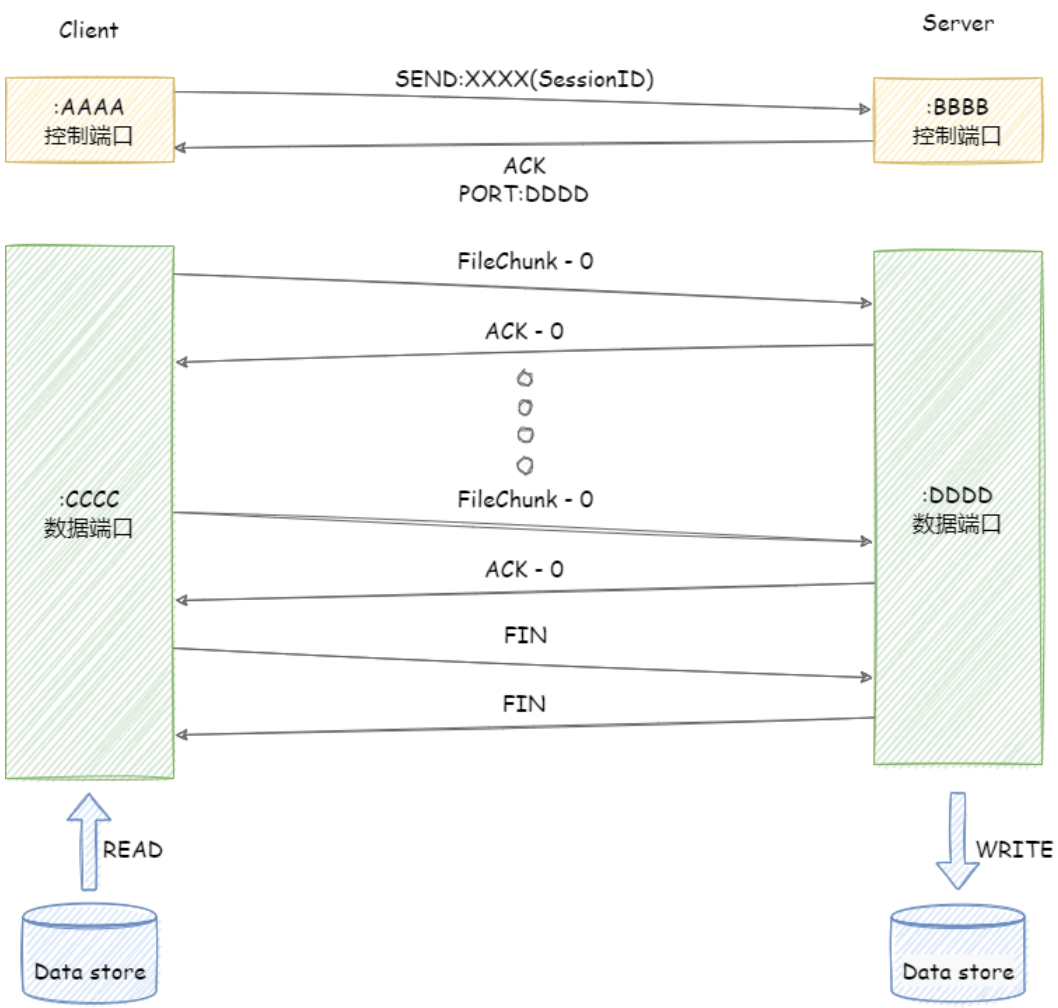
4 整体流程图

整体流程图

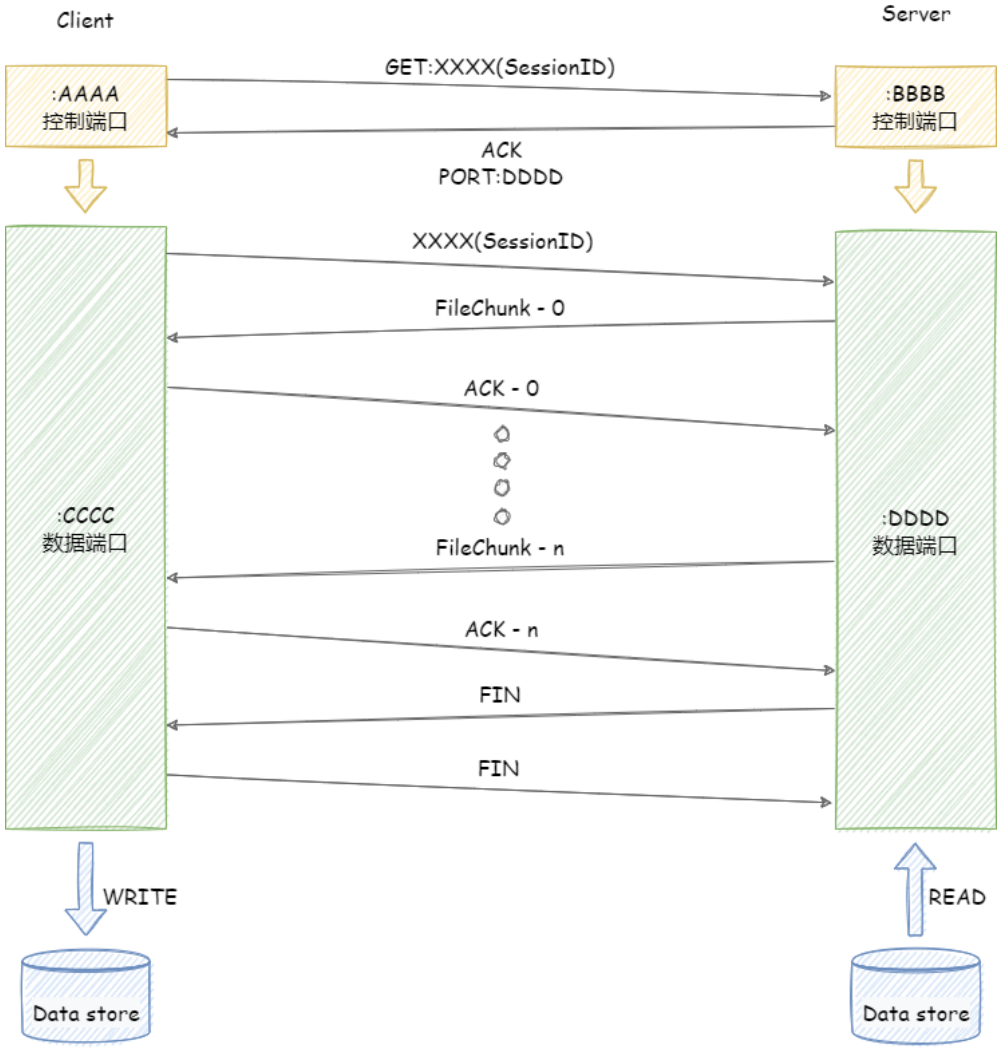


5 上传、下载流程图

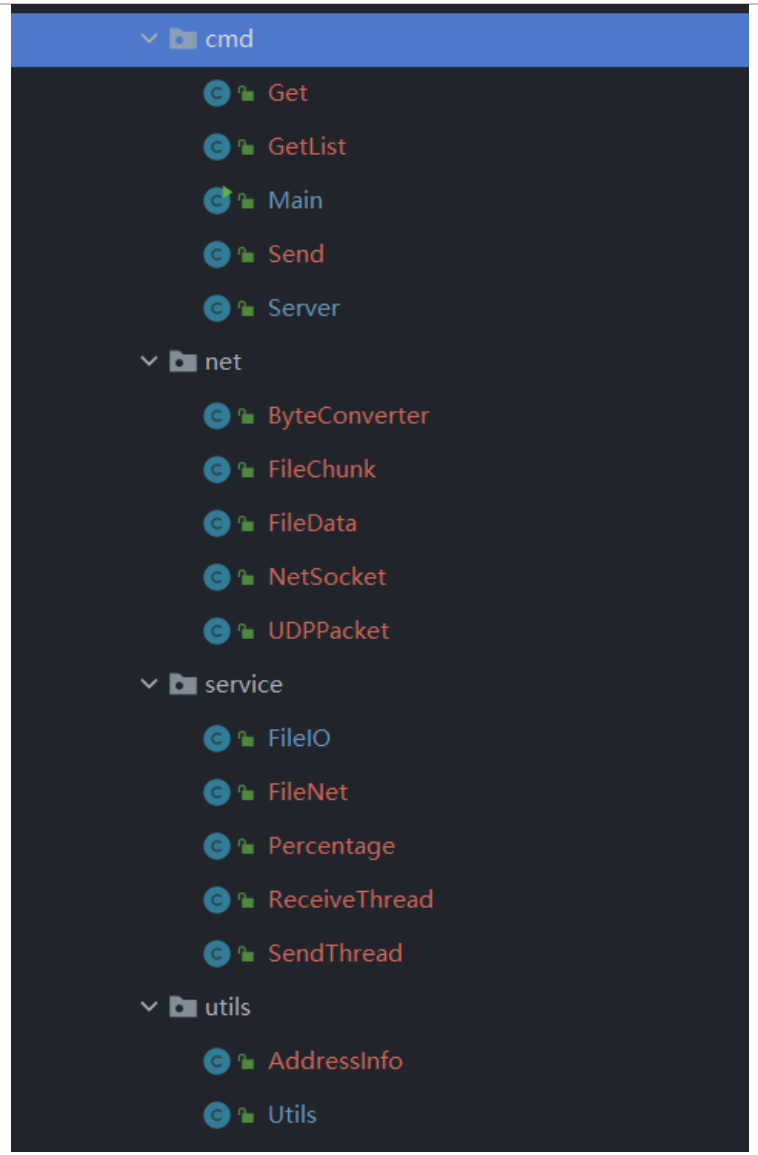
Send file to server



Download file from server



6 简单演示



- cmd // 主要用于CMD命令解析
  - Get.java // 从服务器下载文件
  - GetList.java // 从服务器获取文件列表
  - Main.java // 程序主入口
  - Send.java // 向服务器上传文件
  - Server.java // 开启服务器
- net // 网络相关包
  - ByteConverter.java // Obj、字节数组、Stream转换
  - FileChunk.java // 文件分块及封包
  - FileData.java // 文件块管理
  - NetSocket.java // 网络通讯管理
  - UDPPacket.java // UDP封包
- service // 底层相关实现
  - FileIO.java // 文件IO管理
  - FileNet.java // 文件收发管理
  - Percentage.java // 进度条显示
  - ReceiveThread.java // 接收文件线程
  - SendThread.java // 发送文件线程
- utils // 相关工具类
  - AddressInfo.java // IP地址类
  - Utils.java // 常用工具方法封装

6 简单演示

客户端发送文件：

```
E:\java\middle-ware\target>java -jar MiddleWare-1.0-SNAPSHOT.jar lsend -s 192.168.126.134:3000 javafx.zip
2022-05-30 01:04:49.890 [main] INFO  com.middle.lftp.utils.Utils - Sever IP address: /192.168.126.134:3000
2022-05-30 01:04:49.895 [main] INFO  com.middle.lftp.net.NetSocket - Listening in port 9000
2022-05-30 01:04:49.928 [Thread-1] INFO  com.middle.lftp.utils.Utils - Got send port: 20480
2022-05-30 01:04:49.928 [Thread-1] INFO  com.middle.lftp.net.NetSocket - Port 9000 Closed.
2022-05-30 01:04:49.936 [Thread-1] INFO  com.middle.lftp.net.NetSocket - Listening in port 9001
2022-05-30 01:04:49.947 [Thread-3] INFO  com.middle.lftp.service.FileNet - Start to send file: javafx.zip (39.36MB)
0.74%   [=>                                                                                                ]  292.96MB
1.49%   [=>                                                                                                ]  7.51MB/s
2.23%   [==>                                                                                               ]  6.51MB/s
2.98%   [===>                                                                                              ]  5.63MB/s
3.72%   [====>                                                                                             ]  5.21MB/s
4.47%   [====>                                                                                             ]  4.76MB/s
5.21%   [=====>                                                                                            ]  4.37MB/s
5.95%   [======>                                                                                           ]  4.36MB/s
6.7%    [=======>                                                                                          ]  4.36MB/s
7.44%   [=======>                                                                                          ]  4.32MB/s
99.99%  [=============================================================================================]  6.18MB/s
99.99%  [=============================================================================================]  6.18MB/s
100.0%  [=============================================================================================]  6.18MB/s
100.0%  [=============================================================================================]  6.17MB/s
100.0%  [=============================================================================================]  6.17MB/s
        in 6.38s
2022-05-30 01:04:56.505 [Thread-22874] INFO  com.middle.lftp.service.FileNet - Transmission finished!
2022-05-30 01:04:56.507 [Thread-22875] INFO  com.middle.lftp.net.NetSocket - Port 9001 Closed.

E:\java\middle-ware\target>
```

6 简单演示

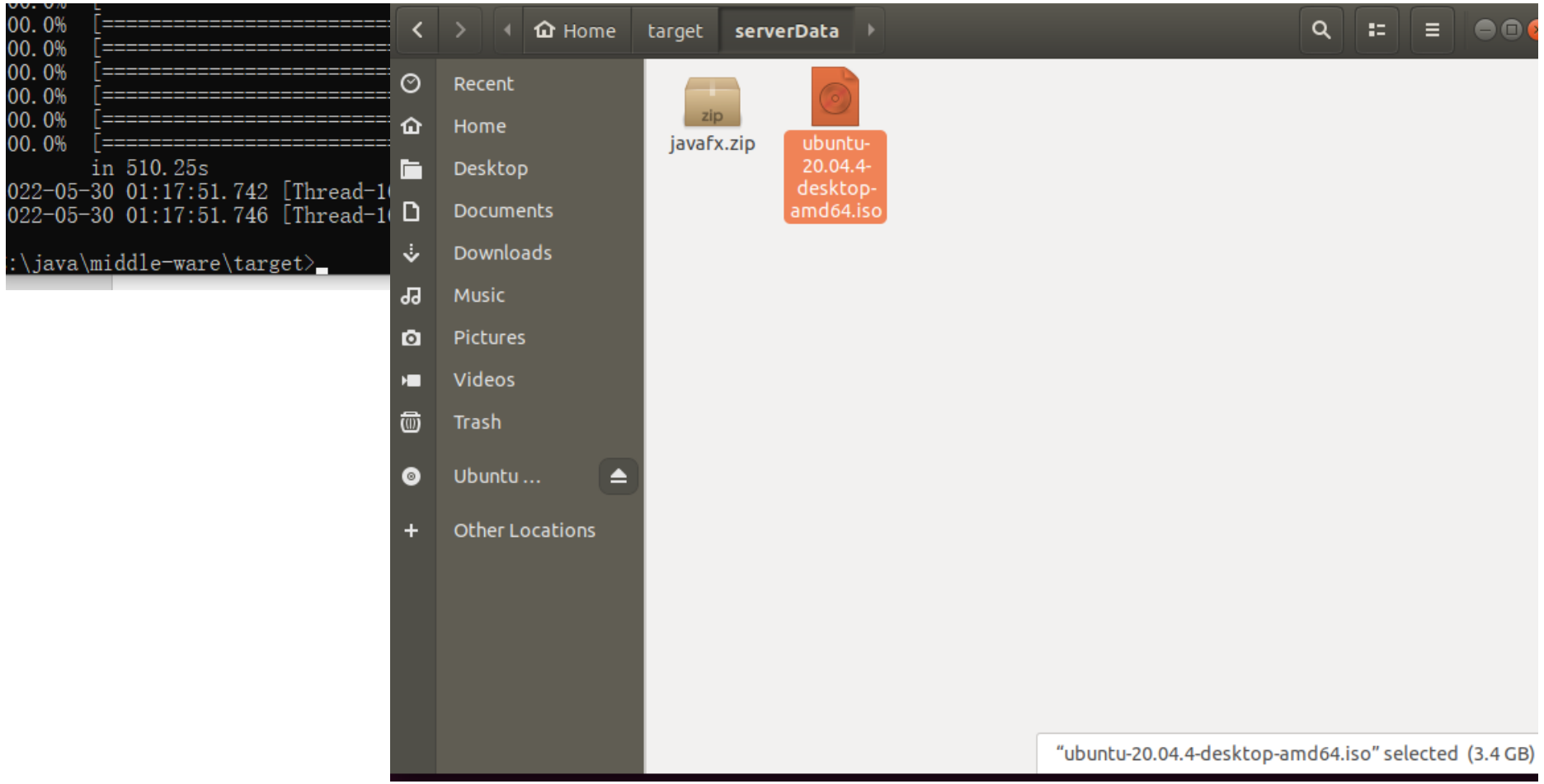
大文件传输：

```
E:\java\middle-ware\target>java -jar MiddleWare-1.0-SNAPSHOT.jar lsend -s 192.168.126.134:3000 ubuntu-20.04.4-desktop-am
d64.iso
2022-05-30 01:09:21.393 [main] INFO  com.middle.lftp.utils.Utils - Sever IP address: /192.168.126.134:3000
2022-05-30 01:09:21.398 [main] INFO  com.middle.lftp.net.NetSocket - Listening in port 9000
2022-05-30 01:09:21.414 [Thread-1] INFO  com.middle.lftp.utils.Utils - Got send port: 20480
2022-05-30 01:09:21.414 [Thread-1] INFO  com.middle.lftp.net.NetSocket - Port 9000 Closed.
2022-05-30 01:09:21.418 [Thread-1] INFO  com.middle.lftp.net.NetSocket - Listening in port 9001
2022-05-30 01:09:21.424 [Thread-3] INFO  com.middle.lftp.service.FileNet - Start to send file: ubuntu-20.04.4-desktop-am
d64.iso (3.15GB)
0.01%   [>                                                                                                    ]  292.97MB
0.02%   [>                                                                                                    ]  8.49MB/s
0.03%   [>                                                                                                    ]  6.42MB/s
0.04%   [>                                                                                                    ]  5.95MB/s
```

```
2022-05-30 01:04:56.379 [Thread-0] INFO  com.middle.lftp.net.NetSocket - Port 20480 Closed.
2022-05-30 01:09:21.375 [main] INFO  com.middle.lftp.service.ReceiveThread - Start receive file
2022-05-30 01:09:21.376 [Thread-1] INFO  com.middle.lftp.net.NetSocket - Listening in port 20480
2022-05-30 01:09:21.387 [Thread-1] INFO  com.middle.lftp.service.FileNet - Receive File: ubuntu-20.04.4-desktop-amd64.iso Size: 3.15GB
```

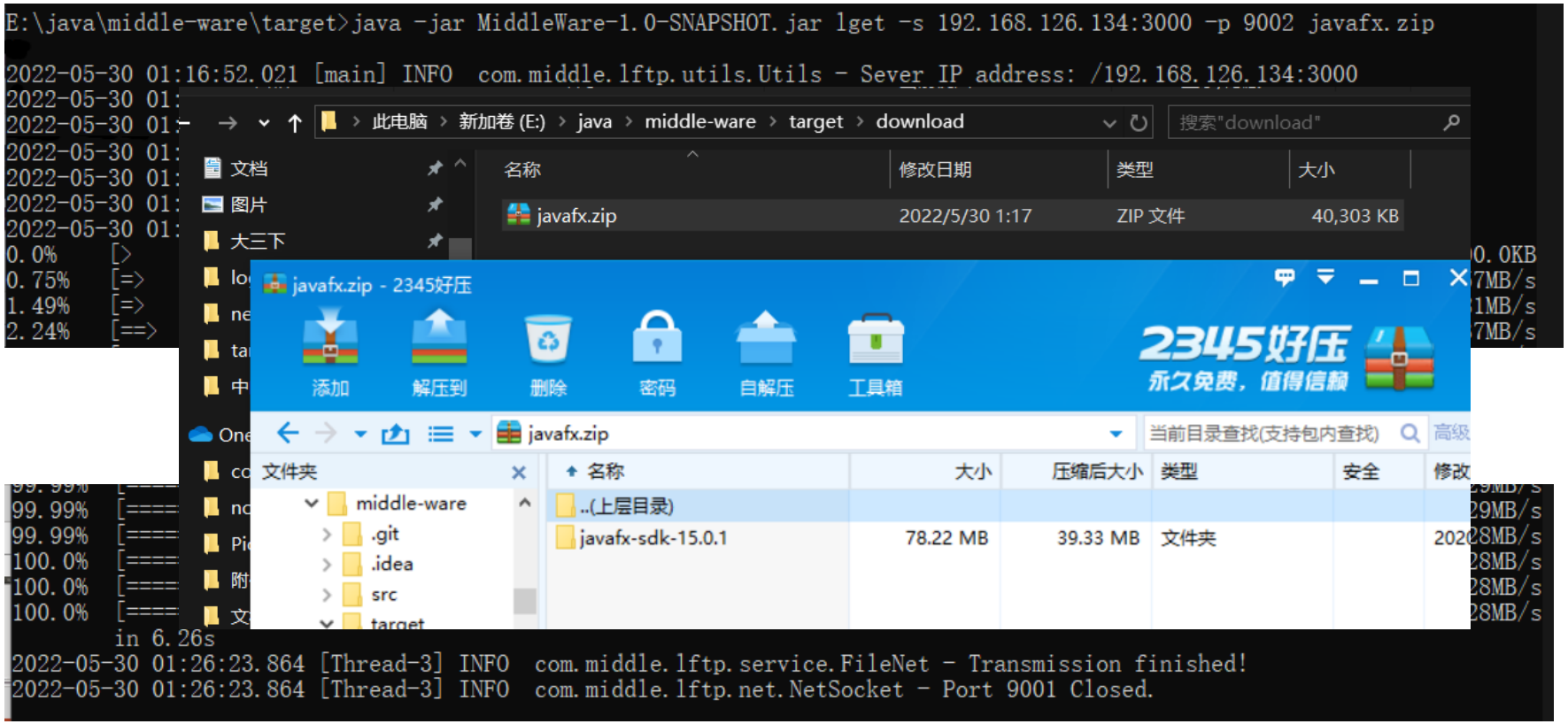
6 简单演示

大文件传输：


00.0%
00.0%
00.0%
00.0%
00.0%
00.0%
in 510.25s
022-05-30 01:17:51.742 [Thread-1
022-05-30 01:17:51.746 [Thread-1
:\java\middle-ware\target>
Home
target
serverData
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Ubuntu ...
Other Locations
zip
javafx.zip
ubuntu-20.04.4-desktop-amd64.iso
"ubuntu-20.04.4-desktop-amd64.iso" selected (3.4 GB)

6 简单演示

服务端接收文件：



7 可能的未来改进方案

- 与ActiveMQ的BlobMessage传输结合
- 网络中断纠错机制
- 断点续传
- GBN → SR （选择重传）
- 大文件传输时存在的传输速率波动
- 公网部署