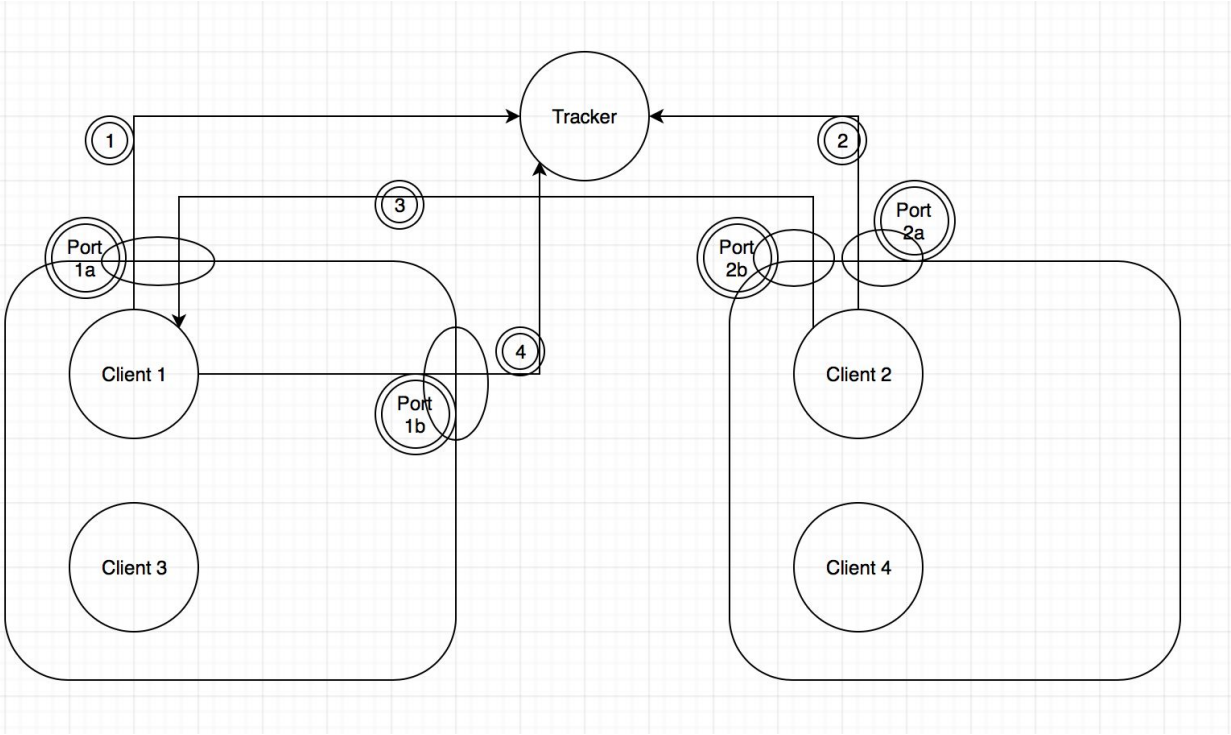
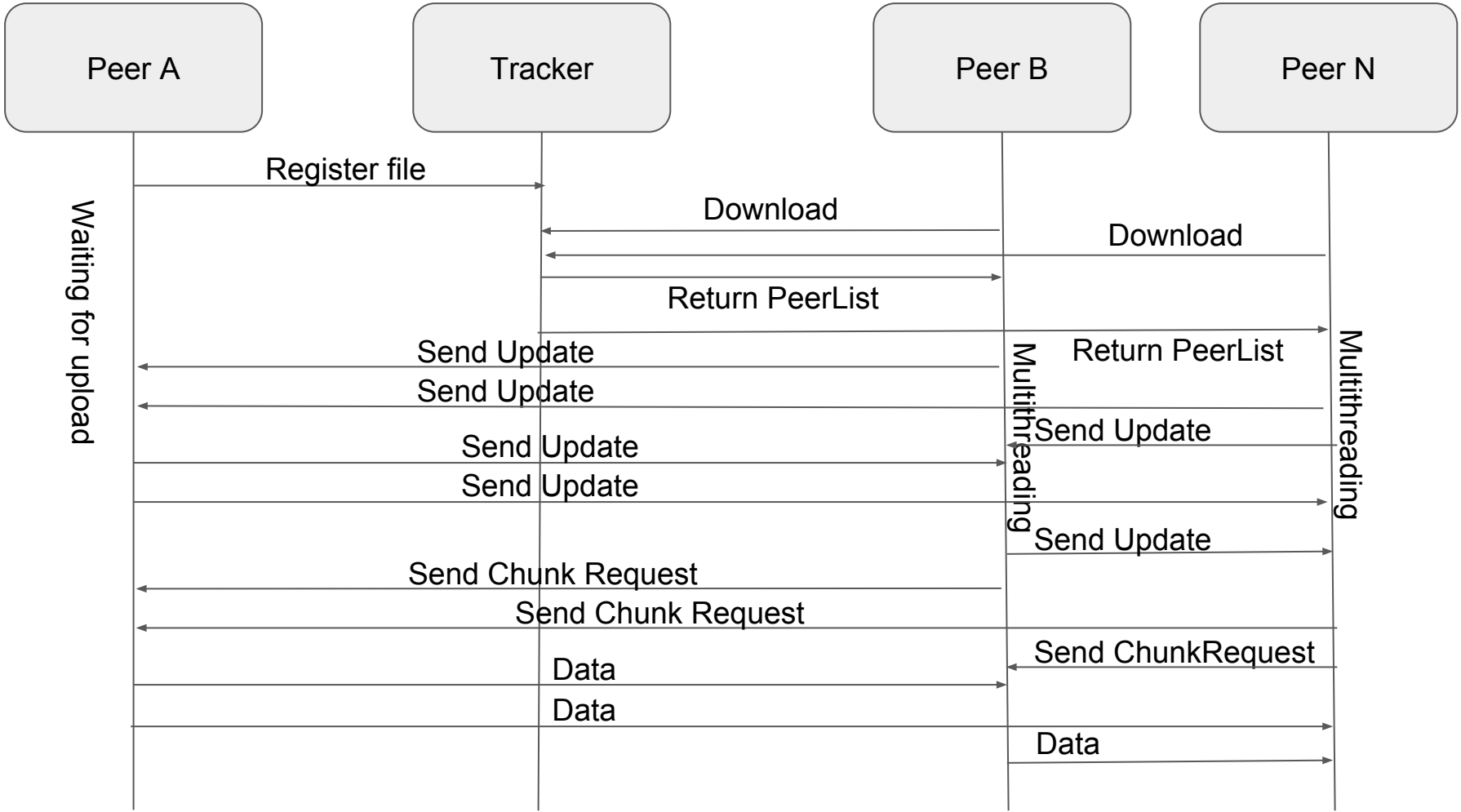
Final Protocol (Group 5)

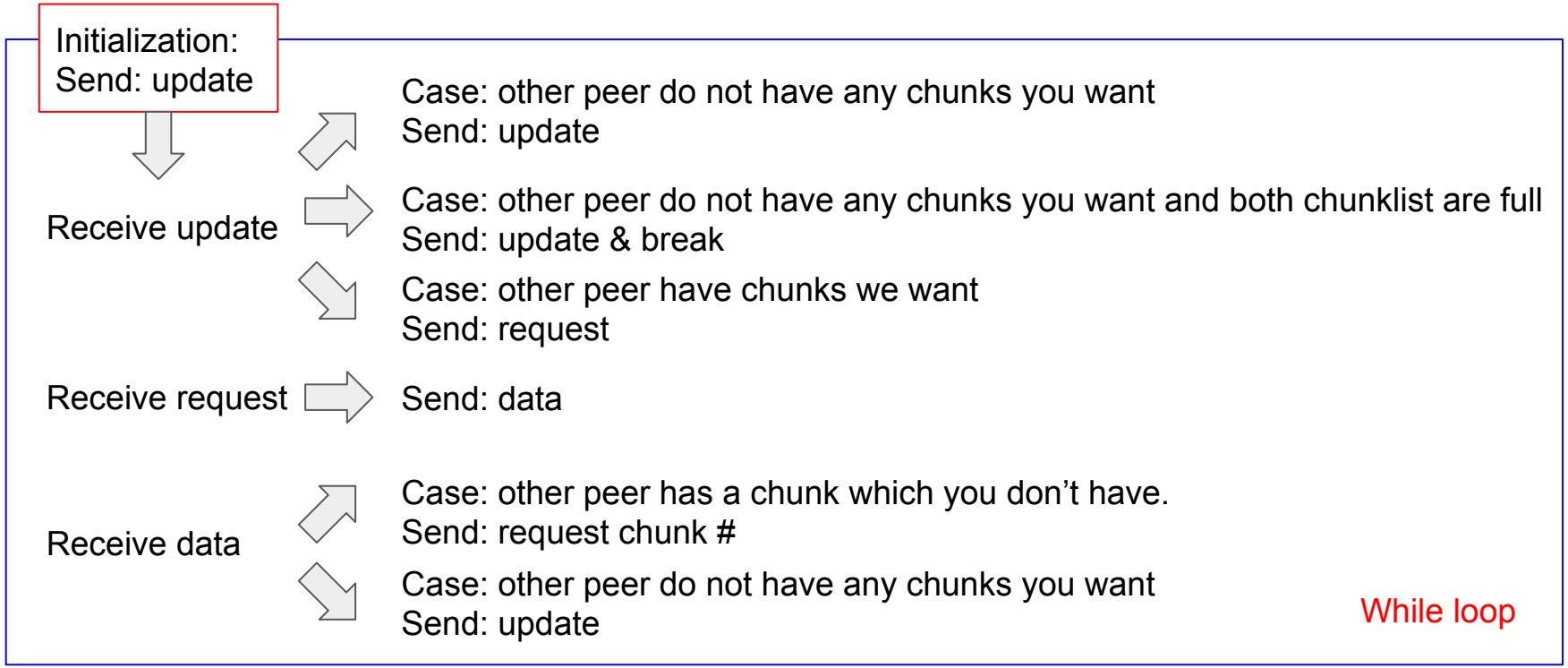


1. Client 1 UPLOAD, inform Tracker about the file. Tracker records Client1's IP and port.
2. Client 2 DOWNLOAD, Tracker returns the list of peers (IP + port) that have that file.
3. Client 2 connect to Client 1, using port 1a which is previously recorded by Tracker. Client 1 responds, and both clients start transmitting data.
4. Immediately after Client 1 receives packet from Port 1a, Client 1 opens a new Port 1b to connect to Tracker. Tracker look at Client 1's peerID to determine if the record already exists, and overwrite Client 1's port information.

- Client 1 will be standing by at Port 1b for new connections.
- Client 2 will be standing by at Port 2a for new connections.
- Tracker has forgot about Port 1a.
- Tracker knows nothing about Port 2b.
- Each connection is one Thread by itself.


Peer A
Tracker
Peer B
Peer N
Register file
Download
Download
Return PeerList
Send Update
Return PeerList
Send Update
Send Update
Send Update
Send Update
Send Update
Send Chunk Request
Send Chunk Request
Send ChunkRequest
Data
Data
Data
Waiting for upload
Multithreading
Multithreading

Logic: (in a thread) one thread/socket



Scenario

Seeder will inform the tracker it has a new file, as well as its self generated peerID. → Tracker will then update its peerID list which maps peerID to the host information (such as ip address and port #) and also updates the fileList which maps fileNames to peerIDs. Seeder will then wait for connection.

Peer 1 wants to download a file from the seeder. Peer 1 contacts the tracker for information of the file. Receive peer 1 info. Use the same port to send update message to seeder (twice). Peer 1 will then create a new UDP port and update tracker of this new port so that other peers can connect to this port. Seeder will receive the update message from peer 1 and because it has nothing to request, it will send an update message back.

Upon receiving the update message, peer 1 will check the chunklist of the seeder with its own chunklist and request for a chunk. Upon receiving the request, seeder will send the chunk over. Peer 1 will receive the chunk and request for another piece.

Get Desired Chunk ID

To reduce the chance of sending duplicate requests

- First Random a ChunkID from our Complete Chunklist as start point.
- From the starting point, return next clear bit as unreceived ChunkID
- If there is no next clear bit, mark the start point as end point and get the next clear bit from index 0 to start point.

Running the program

1) Compile Tracker.java, Client.java, TrackerMessage.java, FileInfo.java, PeerInfo.java by keying in javac x.java (replacing x with the filename)
2) Run Tracker by keying in java Tracker.java
3) Run Client by keying in java Client.java
    a) Proceed to use the interface described in the next slide.

ReadMe

1) Change tracker's IP address at Client.java line 400

2) Initialize Tracker and Client

3) Client will have an interface :

4) To display list of file available for download, enter 1

5) To show the file details like file size, enter 2

6) To download, enter 3 and file name

7) To upload, enter 4 and filepath

8) To exit (at any point in time), enter 5