RUBitTorrent

Phase 3

Richard Gerdes, Rahul Purwah , Robert Williams

Overview

Our BitTorrent is a simple program that dabbles into BitTorrent Protocol. Each class works together as a well-oiled byte processing machine. Entering through the main method found in RUBTClient, we create 3 classes that will lead us through our expedition into BitTorrent: PeerManager, TorrentHandler, and Tracker. The Tracker keeps track of the HTTP posting and pulling. It gets our peers for us and passes them to the PeerManager. PeerManager works to keep track of all of our peers and keeps all socket connections. TorrentHandler controls which piece of the file is downloaded next, as well as saving the file. The small components work like little black boxes for the above three classes, performing dedicated tasks to keep the programing running properly.

The Classes:

RUBTClient

The RUBTClient starts the program, initializes the TorrentHandler and the PeerManager and then begins listening for user input: q and d. If a user presses q the program will gracefully exit. The program initiates downloading with its peers when d is pressed.

TorrentHandler

The TorrentHandler controls everything about the file that will be saved to the disk. It has methods that intelligently decide which piece should be downloaded next based off of a rarest first approach. The TorrentHandler saves any pieces downloaded by the peer.

PeerManager

The PeerManager runs a few different threads, the Server thread listens for incoming connections, validates them and then passes the connection into the list of peers. The PeerListener thread constantly tries to connect to peers given to it by the tracker. Any connection is passed into the peer list (which is synchronized to avoid conflicts). PeerPulse checks to see that a peer is still alive and connecting. If a peer's pulse is gone it removes the peer from our peer list. Lastly we have the ChokeCheck Thread which performs out choking and unchoking.

Tracker

The Tracker manages the HTTP tracker. It performs the posts of uploaded, downloaded, and obtains lists of available peers.

Peer

The Peer class is a thread that transmits data to and from a Peer. It works with Message and TorrentHandler to download and upload. The Peer has two constructors to be used by the threads in the PeerManager class, as well as a few duplicate methods to be used

with each. The Server uses all duplicate Peer methods with a char boob parameter. The parameter is simply to differentiate the methods. The Peer detects error messages sent. IF three error messages are detected in a short amount of time, the Peer will kill itself.

Message

Message produces the messages that will be sent out to Peers. It also validates handshake messages.

Piece

Piece is a wrapper for the pieces of Byte Data

Block

Block is a wrapper for the individual blocks of the Piece

Feedback

We feel that this is a very good semester long project. However, while we worked in a team of three, it always felt like there was never enough work to evenly split between three people. This project may be better suited towards two person teams.

In the beginning, understanding the BitTorrent protocol was very difficult. We think that a tiny bit more could be done to assist in describing the protocol. Something like a diagram that breaks down how the messages are built visually may have assisted in our understanding. Also, stating that the messages would be sent as byte arrays and not as strings would have been convenient.