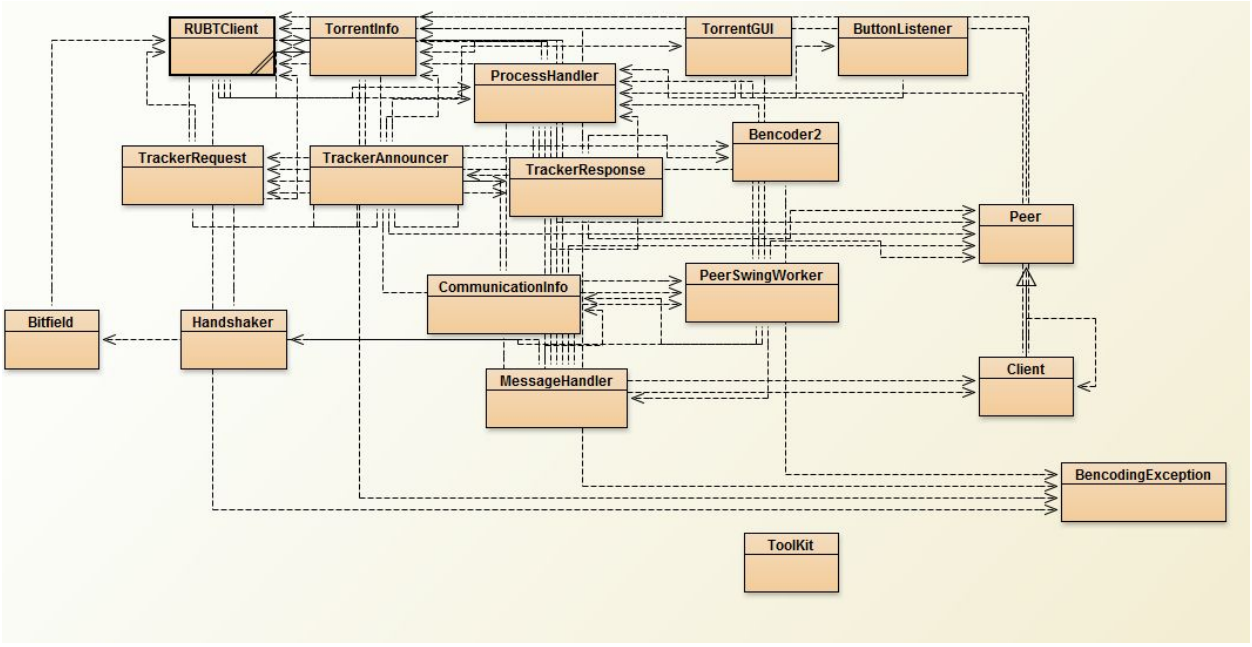
Shuhan Liu (sl1041) 154007082

Nicole Heimbach (nsh43) 153002353

High Level Description:



The main class is the RUBTClient class. It receives two command line arguments. One is the torrent file, and the other is the file we save what we download. If that file already contains part of the download file, this program will count pieces that has been download and continue downloading start from the next piece. It parses the torrent file and then implements the process handler. The process handler will send the request to the tracker by calling trackerRequest, and use tracker response to analyze the information get from the tracker to get the peer list. Then the process handler opens threads with every peer and call the “PeerSwingWorker”. In PeerSwaingWorker, by calling the “handshaker”, it handshakes with peer and verifies the handshake received. Then it opens IO Stream and communicates with the peer by calling the “MessageHandler”. “MessageHandler” exchanges information, downloads, and uploads with the peer.

The TrackerAnnoucer is used to update information with tracker, and bitfield is used to keep track of and manage bitfield information. During the whole process, user can stop whenever he wants by clicking the button created by “TorrentGUI” and “ButtonListener”.

Detailed information of every class is in the following section.

Description of classes:

Bitfield:

Shuhan Liu (sl1041) 154007082

Nicole Heimbach (nsh43) 153002353

Bitfield class can keep track of and manages information related to bitfield value. It can return the bitfield value by calling method get(int index).

RUBTClient:

The main class of this program. It accepts two command line input. One is the torrent file's name and the other is the name of the file user want to store the data in. It will check for valid command line arguments and parses the torrent file. If the file already exists, it will continue downloading from where the last time stops. Otherwise, it will just create a new file and start download.

TrackerRequest:

TrackerRequest class keeps track of and manages information related to sending a request to the tracker. It can send the Http Get request to the tracker.

Handshake:

Handshake class keeps track of and manages information related to handshaking. It will build up the handshake message and send handshake message. Moreover, this class also has method "getHandshakeMessage" that receives handshake message from the peer, and then it verify the handshake message received from the peer by calling "validateHandshake".

TrackerAnnouncer:

TimerTask object that manages periodic tracker announces. It updates information to the tracker, include things like Downloaded, Uploaded, Event, Interval, MinInterval etc.

ProcessHandler:

Keeps track of and manages information used for downloading, including torrent info, the downloading file, timers, and threads. It controls the start and stop of the download process, and it schedules timer to do each task. Also, it creates new threads to handle each connection with other peers.

TrackerResponse:

The same as its name, this class's function is to deal with the response get from the tracker. It will receive the response and then decode the response from the tracker by calling the method "decodeResponse". After decoding, it can get the entire peer list and stores the peer list in an ArrayList, and it also analyze the key sets received from the tracker.

CommunicationInfo:

CommunicationInfo is just a class to build up and object of the information needed for communication with the peer, including request index, download offset, file length and message ID etc.

MessageHandler:

MessageHandler is a core class of this program. It used to deal with the message received from peers and decide to make relevant actions according to the message ID. It can send messages like choke, unchoke, interested, uninterested, request, and have etc. Thus it can handle both the download and upload tasks.

TorrentGUI:

A GUI that allows the user to click the button to exit the program and continue to download later. It will creates a window with a button with text "Exit and continue next time".

PeerSwingWorker:

Shuhan Liu (sl1041) 154007082

Nicole Heimbach (nsh43) 153002353

PeerSwingWorker is just a thread function that deals with connection with every single peer. It connects to peer by calling function ”connetToPeer”. In that function, it will send the handshake message to the peer and verify the handshake message received. If the message is valid, then it will open the IO Stream with the peer and then communicate with the peer. It send and receive messages from the peer and then call “MessageHandler” to deal with those messages.

ButtonListener:

This class is used when the button in the GUI is clicked. The resulting operation of the button is determined by the current state of the program.

Client:

Client class extends the “Peer” class, but it’s for building ourselves as a peer. It sets up Boolean flag “downloadComplete”. It generates our own peer ID.

Other classes are given by the assignment.

Who did what:

- Nicole Heimbach
  - Javadoc
  - Organization of code into classes
  - Implemented most of the things relating to tracker announcing
  - Implemented simultaneous uploading/downloading (multithreading)
- Shuhan Liu
  - Handshake with the peer
  - Implementeed most of MessageHandler (including most of clientDownloadPiece(), peerRequest(), and sendRequest)
  - Fixed the problem where the program could not stop after finishing the download
  - Implementation of ability to restart download after exiting the program