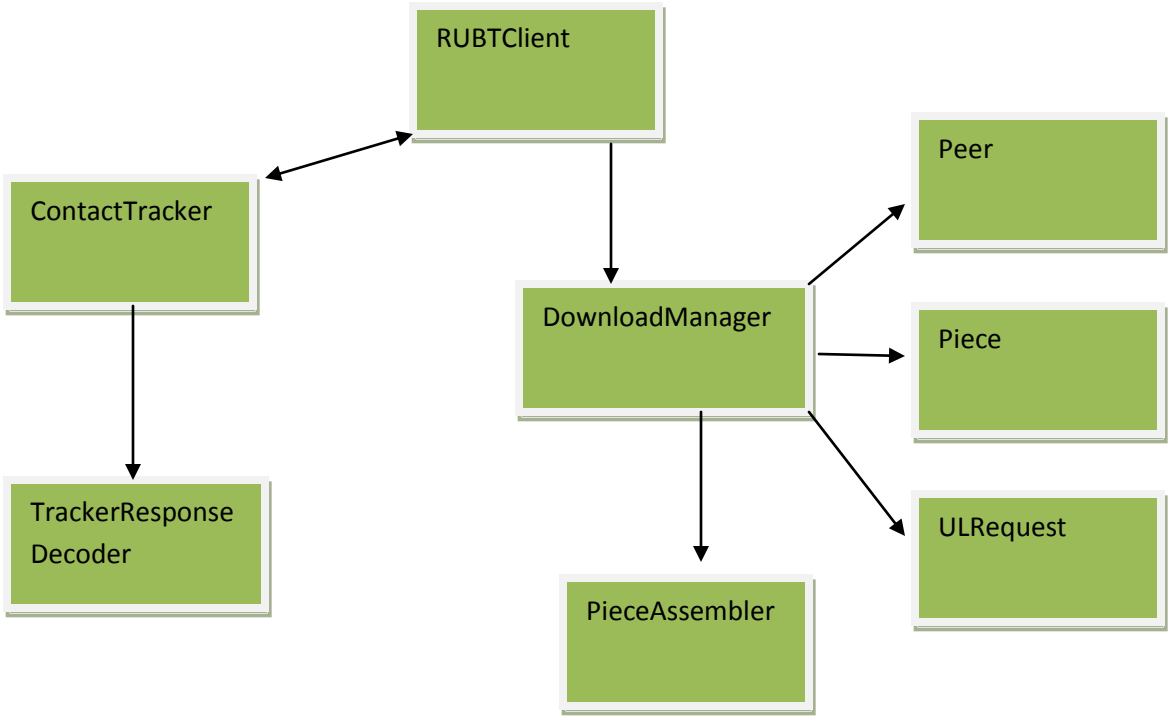
RUBTClient Project - Part 1

Kyle Waranis - kwaranis
Thomas Travis - tptravis
Yuriy Garnaev – yurgar

Overview -
The general dependencies of our Bittorrent client can be best described by the following diagram:



In general what our program does is generate a number of worker threads to handle all parts of the Bittorrent process, while trying to keep them segregated where necessary. The first thread that spawns is the ContactTracker thread which manages all communication with the tracker, including the regular notifications. Then it creates a DownloadManager that starts the process of connecting with peers and downloading/uploading. Within the DownloadManager one thread is maintained for downloading, one for uploading, one for accepting incoming connections, and one per peer for accepting communications from peers. Finally, a thread for user input is created that controls the applications state based on the users input. A download can be paused, resumed, stopped, and queried for its download state.

Control Classes:
RUBTClient.java - This is the class containing the main method. It starts communication with the tracker and then creates and starts the DownloadManager. Once all of the worker threads and classes have

been started it loops for user input and controls the general state of the program accordingly. The downloaded file is saved in the /Downloads folder inside the project folder.

ContactTracker.java - This class communicates with the tracker to retrieve the list of peer URLs and other relevant information. It opens a URL connection to the tracker and uses the TrackerResponseDecoder to retrieve the information. It does this at an interval defined by the tracker.

DownloadManager.java - Download Manager (DM) keeps and maintains a list of peers that it walks through the connection and download process. In addition, the DM class handles the coordination between the information received from the peers and the PieceAssembler. In addition, DownloadManager.java contains the Piece and ULRequest classes which are used as an abstraction tool to simplify the process of recording and saving data.

Peer.java - The Peer class is an abstraction for the Download Manager that provides an easy means to communicate with any given peer. It is responsible for assembling and sending messages to the peers, receiving and processing remote peer's instructions, and keeping the connection with the remote peer open.

PieceAssembler.java - This class manages the logistical matter of storing and eventually saving to a file the pieces that are received from the peers. The Piece assembler tracks the received blocks and matches them to their corresponding indexed piece. When a piece is complete its return values will notify the Download Manager which can then send 'have' messages. Once the entire file is completed it will write the received data to the requested location.

Resources Classes:
TrackerResponseDecoder.java - This class un-Bencodes the tracker response byte array, storing a list of peer URLs in a string array, and any other key-value pairs returned by the tracker in a hash map. Used by the ContactTracker class.

TorrentInfo.java - Provided by Robert Moore II. The TorrentInfo uses the Bencoder2 class to un-Bencode the torrent information in the .torrent file.

Bencoder2.java - Provided by Robert Moore II. Used by TrackerResponseDecoder to un-Bencode byte arrays.

BencodingException.java - Provided by Robert Moore II. Exception thrown by Bencoder2.java.