- A description of your implementation of BitTorrent (1 page)

Bit-tortoise starts off by generating our peerID. Then, it parses the .torrent file. From the information in the torrent file we create a list of pieces and blocks inside the pieces. The purpose of the list is to organize what blocks we will later request and receive, to keep track of how much of a file we have gotten. The next step is to send an HTTP GET request the tracker to obtain a list of peers. Once we get the response from the tracker, we parse the becoded data, and form a list of Peer objects representing the peers ip addresses and ports. From the list of peers we open connections (using SocketChannels) and add them and a ServerSocketChannel to a Selector, keeping a Mapping between the SocketChannel and its respective peer. We loop waiting for the selector to tell us when we have received data or are able to send data over one of the channels. When a Channel is ready to be read, we read and parse the message, changing state in the peer object. When a Channel is ready to be written to, we check the state of the peer object, and send a message if necessary. Whenever we receive an unchoke message we request a random available piece the peer has.

- A description of your extensions (if any) (1 page)

- A detailed list of work done by each project member (1 page)

Work Done by Members:

Andrew: Setting up SVN server, wrote deprecated receive message code. Wrote code to ping the tracker and process response including storing the peerlist. Abstracted code into tracker class. Creating Sha-1 Hash for given data. Handling code for unchoke messages.

Rob:

Will:

Kenny:

- Description of experiments and results (no page bound)

- for each experiment, please state the hypothesis you were testing, and the results you observed, and your conclusions.