Design Document

Basic Theory:
We use Java RMI to implement the communication between Peers. Since the RMI has already realized the thread for each peer client communication with peer server, we only need to create threads in Peer, where thread “peerClient” is used for showing menu for the user and thread “executeQuery” is used for running query. Additionally, we add two new threads this time: the update thread is used for monitoring file last-modify-time and update the version when time changes, and it can also be used in push mode; the PullMode is used for pull mode to automatically check and update.

There we list each code with design description. Since there exists old function from PA2, we only list the new design or change on the list:

1. **PeerI.java**

The Remote interface for Peer. We add two new method signature: invalidation() and pull().
Moreover, we add three sub classes for packaging file information:
FileInfo: record version number and last-modify-time, used for master copy.
CachedInfo: additionally record TTR, origin Peer IP, file state and cached time, used for cached copy.
FilePackage: additionally record file date, used for obtain (download).

2. **Peer.java**

The new or changed code for one Peer:
Attributes:
- msgHit: Instead of just record one string peer IP, it record IP, last-modify-time and state this time, so the value become “String[]”.
- originFile: This map records the key of master copy filename with value of the file's info, such as version Number, modify date, consistency state.
- cachedFile: This map records the key of cached copy filename with value of the file's info, such as version Number, modify date, consistency state.
- invalidmsgToPeer: This Map records the key of invalidating message ID with value of its upstream peer ID. It's used for spread invalidating.
- Invalidmsglist: The Queue maintain the sequence of invalidating message. when its size is invalidmsgsize and new massage is coming, the oldest one will be removed.
- Invalidmsgsize: maximum size of Invalidmsglist.
- TTR: The default time-to-refresh. The dimension is second.
- cachedfilepath: The relative path where saves those cached files.
- mode: Record the mode of push or pull. Default is None (both off).

Methods and classes:
- query():
  - Instead of checking the file in local path, we check the originFile map and cahcedFile map, if the file is found, it will call queryhit with corresponding file information.
- queryhit():
  - Add new parameters: FileInfo info is used to record states of file; boolean origin is indentify where this file is master copy in the peer.
  - Instead of just record Peer (server) IP, we record more information which are last-modify-time and whether origin. This records will be shown when obtaining file.
- obtain():
  - Return more info about the file, which is recorded in object filepack.
- invalidation():
  - Mostly like the query method.
  - This method receives the invalidation query call from upstream peer. The messageID contains both the real message ID and the upstream peer Url, so firstly it will be splited. Then we should check whether this message has

been queried to this peer before by checking invalidmsgToPeer. If it contains, we just skip this query(), otherwise we will maintain the invalidmsgToPeer and check whether this peer contains this file with "filename" in cachedFile map. If the map keeps this file, it will check the version and change the state. If the TTL is over 1, this peer should spread the invalidation() to other neighbors with TTL-1, besides the upstream peer.

- pull():
  - The pull function simply check whether the versionNo is the latest one.
  - If versionNo is same, return fresh TTR. Otherwise, return 0. Additionally, if server cannot find file in list, return -1.
- peerClient:
  - This thread class will show the menu for users and supply corresponding methods.
  - This time it add four new cases for setting TTR, setting mode, modify file and show copies info:
    - Option 6: Simply change TTR value;
    - Option 7: Simply change mode type; For pull mode, it should start new PullMode thread or interrupt it according to the selected mode.
    - Option 8: Simply change last-modify-time of specific file.
    - Option 9: Call showCopyInfo() to print out all the info for both master copies and cached copies.
  - It add two new thread classes and four more functions:
    - recordFile(): Record the info of files under the filepath into the map of originFile. In other words, record the info of master copies.
    - Update: It is the thread for updating origin file states, mainly change the modify time and version number. Simply call update() in every two seconds.
    - updateRecord(): This method will update the info for master copy files (in originFile). The info contains version number, modify time. Additionally, in push mode, it will call invalidation to the neighbors if the last-modify-time of master copy was changed.
    - PullMode: In every two seconds, the thread will check whether the file in cachedFile should be TTR expired. In every thirty seconds, for each TTR expired file, the thread will call push to origin server to check whether file is really invalid or still valid.
    - changeModifyTime(): Ssimply get the file with the filename from user, change the last-modify-time to now.
    - showCopyInfo(): Traversal the originFile map and cachedFile map, pirnt out the information recorded in it with corresponding filename.
  - One function has been updated a little:.
    - executeDownload(): This it will not only download the file data but also get the information of this file. This information be saved into cachedFile map.
- main(): It creates a Peer object, registers RMI on specific port (e.g. 1101, each peer keep different port), and bind the server on specific URL (e.g. rmi://127.0.0.1:1101/Peer). Then it is waiting for other Peers to connect. Additionally, it start a client thread for user to operate.

Possible Improvement:

1) If there exists the situation that two peer keep a same master copy, we may need check which copy is the newest one and also notice which origin server a peer download cached file from.
2) Sometimes the file isn't really modified, under this possibility, we may need check whether the content in the file is changed.