CSE 452/M552
Project 2: Fault tolerant updates by multiple clients

Design document due: May 2, 2013
Project Due: 5pm, May 14, 2013

In this assignment, you are to add support for multiple clients simultaneously reading and writing data to the server, where updates are handled as transactions. Your system should be able to survive any number of simultaneous client and server failures, and as long as the server is up, clients should be able to continue to update follow lists and post tweets.

With a transaction, clients can group operations together, e.g., to add a tweet to a list of files for clients that are following that user. The operations commit together or not at all, in globally sequential order. In the case of an inopportune failure or the transaction cannot be completed without violated sequential consistency, the updates are rolled back to the state before the transaction started, and the client can simply retry the transaction. Depending on your application code, some of your transactions may have only a single read or write inside the transaction; that is of course just fine – the transaction is still either done completely, or not at all.

Transactions provide a clean way to deal with failures in the presence of cache coherence. Although there are several ways to integrate these concepts, one particularly convenient way is called multi-version concurrency control. In this scheme, clients can cache copies of any data they might need, tagged with the version of the data being cached. When another client updates the data, it just bumps the version # on the server. A transaction can only be committed if it is based on valid versions at the server; otherwise it is aborted, and the client can refetch the data and start again.

For this assignment, you only need to handle one transaction at a time per client.

The turn in instructions are the same as the previous assignment.