P3.2 Design Analysis

The current implementation stands as a very similar structure to that of the Object Model. The only differences are that the PostVote and CommentVote objects replace the PostUpvote, PostDownvote, CommentUpvote, CommentDownvote objects. This decision was made because there are only two types of vote (up and down), and thus the difference between them can be represented with the Boolean "up" value. This does not reduce the amount of data storage required for votes, but does simplify the object structure.

The PostVote and CommentVote objects are almost the same (except that they affect different objects). The decision to keep them separate was made because it logically makes more sense to keep them in separate tables rather than have an "iscomment" switch. This also allows for greater extension in the future, as Post and Comment may diverge in their representations in the future.

The only invariant that is nontrivial in the code is that a user may only vote Up or Down for each comment and post. Users may vote multiple times, but only the most recent vote will "count".

Rejected designs include mashing Comment and Post into one table, and CommentVote and PostVote into one table (separated only by Boolean switches). This is to allow for greater extensibility in the future, as described above.

The design issues to be resolved in the future are the nesting of comments functionality. Currently the functionality is built into the models for comments to comment on other comments, but the controllers and views do not yet support this functionality. Similarly the models are able to represent user karma, but karma updating has not been built into the controllers and views. These will be included in phase 3 of the project.

As for the actual implementation of the refresh-free updates, a short-polling method is used that works by having the view request new data from the server every 8 seconds. This is not a fine operation: each request refreshes the entire view and would be cumbersome with large datasets. This *could* be modified to do something a little smarter (only show new posts since the last refresh), but that does not seem necessary within the scope of this project.