Design Notes

Yuqi Liu <yuqil@andrew.cmu.edu>

1. Protocol between Server and User-node

In this project, two-phase commitment is used to commit and publish collages.

- Someone starts a commit process. Server sends out requests to ask all users involved to vote.
- Server waits for at most 3 seconds to collect votes. If all agree, then it replies all users with SUCCESS. If there is one disagree vote or timeout vote, then it replies all users with FAILURE. Server waits until receiving all ACKs. If SUCCESS, post the new collage. If ACK not received, repeat sending every 3 seconds.
- User receives the commit decision and replies with ACK. If SUCCESS commit, deletes local source.

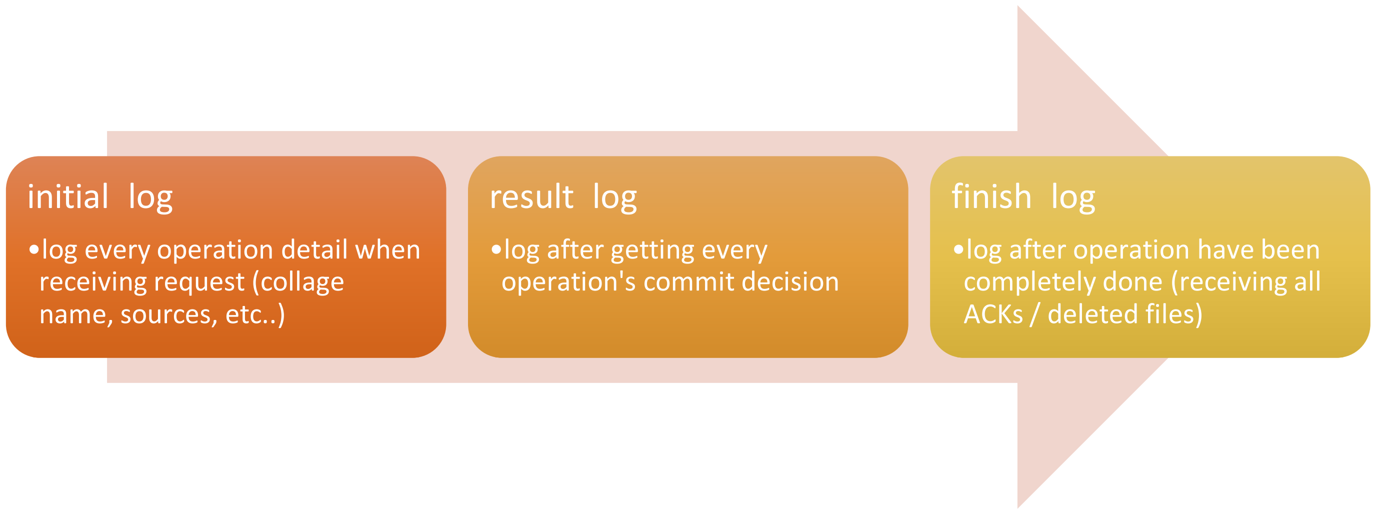
2. Lost Message Handling

There are three message lost cases.

- Lost of users' vote message: then the commit timeouts after 3s and abort.
- Lost of commit decision message: ACK will not be received in this case and server will repeatedly send the decision every three seconds
- Lost of ACK message: server repeatedly sending the decision every three seconds until node's ACK is received.

3. Log File Structure

There are three types of logs in the Log.txt file, Initial log, result log and finish log.


initial log
•log every operation detail when receiving request (collage name, sources, etc..)
result log
•log after getting every operation's commit decision
finish log
•log after operation have been completely done (receiving all ACKs / deleted files)

Each log line can be splited using space. Each type of log is logged as below. Second token denotes which type of log it is.

Initial log: filename log source1 source 2 source 3
Result log: filename result result-code
Finish log: filename finish

4. Failure Recovery Semantics

There are three types of logs per operation. The logs are logged in Log.txt in time order. Every time a node starts, it firstly recovers from Log.txt. All other operations must wait for recovery to be finished.

For server side:

For operation with all three logs, it means the operation needs no recovery.
For operation with initial log but no result log, it aborts the operation and sends the decision to user nodes.
For operation with initial log and result log, but no finish log, it repeats the decision distribution process.

For client side:

For operation with all three logs, it means the operation needs no recovery.
For operation with initial log but no result log, it does nothing. Because by default the operation will timeout so it causes no effect to result.
For operation with initial log and result log, but no finish log, it repeats the process of unlocking the file and remove the file if necessary.

5. Other Design Decisions

(1) The user node will lock all source files needed for a commitment. If following request asks for locked file, it will reply with disagree.
(2) Call back message handling is used in this project. At server side, each commit process has a specific queue for message receiving.
(3) File operation are in sequential order to ensure time order of the log.
(4) Fsync is called after every log writing.