Can Wang

1. Implementation

I finished the MP2 based on my MP1 implementation, so some of the project file name is still "mp1".
Don't worry, I have updated all the corresponding classes to support the requirements in MP2.

First Replica class is inherited from Console class which receives delivered messages from Multicaster.
And Replica class has two sub-class:
(1) LinearReplica which implements the linearizability consistency model
(2) EventualReplica which implements the eventual consistency model
This design really makes the implementation much easier!

Replica should implement the following two method to handle the requests from the clients and the
messages from other replicas in the system:
(1) onDelivery(String id, String message) : the argument 'message' is received from one replica and
this message could be a request or an ACK.
(2) onRequest(String clientId, String cmd, String[] args) : the argument 'cmd' could be 'put' or 'get'
which indicates the type of operation the client is asking. The other arguments of this operation will
be stored in the third arguments 'args'

For LinearReplica, the underlying multicaster is TotalOrderingMulticaster so that the real time order
is preserved, when LinearReplica receives a request from a client, it broadcasts this request to all
the replica in the system, and sends back the response once it receives the ACKs from all the replicas.

For EventualReplica, the underlying multicaster is the basic multicaster, in 'onRequest', it will broadcast the request to all the replicas in the system. It will send back the response to the client
once it collects enough the ACKs from replicas.

2. Usage

First change to the path of binary class:

cd PATH/TO/MP2/out/production/mp1

To launch the replicas:

java com.company.Replica <config file path> <id> <consistency model(linear/eventual)> [W, only for eventual model] [R, only for eventual model]

To launch the client:

java com.company.Client <config file path> <cliend id> <replica id>

And I did this mp by myself.