# Overall Program Design:

Our implementation uses Python and the python xmlrpclib RPC library. The system consists of a gateway, three sensors, two smart devices and a user process. The communication between gateway and distributed clients are done by remote procedure calls (RPC). There are five important RPCs in the system: *register(type,name,localadd), report\_state(devid, state) and change\_mode(mode)* are provided by the gateway*, query\_state(devid), change\_state(devid, state)* are provided by distributed sensors and smart devices, *text\_massage(string)* is provided by the user process.

# Design Details (Lab 2):

Leader election and clock synchronization:

Vector clocks:

Each party participating has a unique index in the vector clocks. We use the device id assigned at registration as the index and set the id of gateway to be 0. Every time a message is sent, sender initiates a multicast to all other parties of the current time vector. On receiving a message, the receiver updates bits of all other parties and add one to its own value. In such way, every message has got a time vector, and the system can provide causal ordering.

Event ordering:

## Back-end database:

The database process works as a back-end tier, and it interacts with the front-end tier with through three interfaces: write (cid, state, timestamp,vector) writes into the database, recording the state, timestamp and clock vector. read(cid, timestamp) reads from the database. When timestamp == 0, returns current state; when timestamp >0, return state of time indicated by timestamp; when timestamp < 0, returns all the state history of the device. read\_offset(cid, timestamp, offset) returns the state of a device from (timestamp – offset, timestamp). The gateway and the database interact through RPC calls.

# How to run the program:

## Deploy on a single machine: (code\_single)

1. The input file name is "test-*input.csv*"
2. Run the bash script file "*code\_single/run-all.sh*". Output will be in separate files. Or run "*code\_single/run-test-case.sh*", output to “code\_single/test-output.txt”.

Note: The clients and gateway uses port number 10000 to 10006 to communicate. To adjust these ports, modify *localadd* values in "*setting.py*" file.

Deploy on separate machines: (code\_distibuted)

1. For the three *setting.py in server,client and user folder*, set *serveradd* to be the ip address and port the gateway uses, set individual address for the remote clients in *localadd*.
2. To solve the time synchronization problem, we use an extra RPC call at port number 10001 to set start time at two different machines to be the same. Modify *syncadd* to change the port.
3. Run "*code/distributed/server/run-gateway.sh*" and " *code/distributed/client/run-remotec.sh*" separately on different machines. Also, since we deploy the user process on the same machine with the remote clients, make sure the client folder and the user folder are under the same directory.

# Test case explanation:

# Possible extension:

### Group member:

Tengyu Sun

Dan Zhang