Hi Prateek,

Here are the build and run instructions.

Building once you have the source code in your go1.4.2 environment:

1. `go get github.com/oleiade/lane`
2. `go get github.com/nu7hatch/gouuid`
3. `./scripst/compile_all.sh`

Running (assumes binaries created are on your path):

1. `testcontroller` is a program that can manipulate the sensor states and instruct the gateway to do some queries. It takes a configuration file and a test file as arguments.
2. The configuration file states the IP addresses and port numbers of all processes, as well as which should be started locally. It also configures the gateway polling interval for temperature sensors. The specification for the database directory for storing files has not be implemented and will default to the current directory where the database process starts.
3. The test file is a list of instructions telling `testcontroller` how to change sensor state, instruct the gateway to query state, or kill and start a replica. The time is given in milliseconds since the start of the test. The states are only used if the command is `ChangeState` and the states are as described in `api/api.go`. Sorry this can be confusing.
4. Example: `testcontroller -c configs/rep_all_local.json -t tests/events_failure_interleave.json`
5. The required processes can also be launched manually and `testcontroller` can be used only to run tests when `StartLocalProcesses` is an empty list, as in `configs/manual.json`. Unfortunately, test controller can only kill it's subprocesses, currently.
6. `testcontroller` can also be used to start processes without running a test by not passing a test file. This allows `testcontroller` to start remote processes first on another machine before another `testcontroller` starts processes and runs instructions.

All tests and outcomes are located in `tests/` and configs are in `configs/`. The design doc is `doc/CS677DesignDocument_final_LAB3.pdf`.

Thanks for your time!