

Programming Project 1

This is the first of two programming projects using the JGroups toolkit. To get started, read and follow the instructions in

<http://www.zurich.ibm.com/~cca/sft08/proj1/README>

We provide two artificially simple “reduced” protocol stacks: `udp_reduced.xml` using UDP and `tcp_reduced.xml` using TCP as basic transport. We need these stacks because the default stacks (these are in `udp.xml` and `tcp.xml`) have large message retransmission buffers, do not time-out fast enough, and have extra functionality that is not needed here. The custom stacks specify small buffers and do not support state transfer between group members.

1 Group Membership

To demonstrate the effects of group membership, run the Draw demo with the protocol stack in `tcp_reduced.xml`. Type multiple times in separate shells:

```
./jgroups org.jgroups.demos.Draw -props tcp_reduced.xml
```

Delay one instance (by typing `^Z` at the shell to suspend the process) and observe how the other instances deliver a new group membership (view) that excludes the delayed one. Continue drawing in the others. Then resume the delayed instance (by typing `f9` at the shell to resume the process) and watch it join the group again.

2 Reliable Delivery and Total Order

The installation instructions mention the Chatter program. Familiarize yourself with the code.

Chatter supports an automatic mode where group members repeatedly send messages containing the current Unix time. Automatic mode is enabled by specifying the parameter `-auto <no_seconds>` with a non-zero argument.

In automatic mode, a member starts sending messages as soon as the group contains enough members, as specified by the parameter `-size <no_members>`. Chatter stops sending after the specified number of seconds given with the `-auto <no_seconds>` parameter. There is also a hard-coded `Thread.sleep()` statement that inserts a 1ms delay between two broadcasts.

Differences between UDP and TCP stacks. Run Chatter in automatic mode with the stacks in `udp_reduced.xml` and `tcp_reduced.xml` and identify differences.

For example, run it like this for three different users:

```
./jgroups Chatter -props udp_reduced.xml -user A -size 3 -auto 10
```

Send the outputs to a logfile and examine the differences between the logs.

Alternatively, you can also extend Chatter to detect if messages are delivered in (per-sender) FIFO order.

Total Order. Total ordering of messages is provided by the SEQUENCER protocol. Insert the SEQUENCER protocol at the top of one of the stacks (that means in the last line for the XML configuration, e.g., as seen in the file `./conf/sequencer.xml`).

Explore differences in the protocol execution with and without the total-order layer! Again, log the outputs and compare them or extend Chatter so that it provides an output from which you can detect the total-order delivery.