Project 5
Due: April 19, 2011 11:55:00PM (Edge 4/21/11)

In your implementation of project 4 using Java multicast facility, you probably have never encountered any noticeable message delay or loss in the protocol (or algorithm) since the multicast is done in a local network domain. In a larger network domain, message delay and loss are non-negligible. This project is a simple extension of project 4 that includes a simulation of message delay and loss in the token-based distributed mutual exclusion protocol. Message delay or loss can be simulated either at the sender or the receiver side through randomization. You are free to choose either approach.

This project emphasizes more on the understanding of the algorithm execution than the coding of your program. Thus, the grading criteria will be different. First, your program should run correctly. Then, we would like to see the discussion of your project in two aspects: 1) how you simulate the delay and loss of messages, and your insight of the implementation, 2) your observation of the effect of delay and loss of messages on the correctness or efficiency of the SKB distributed mutual exclusion algorithms with some output examples.

Your report should be no more than two pages. Provided that your program runs properly, the grading will be relative. We will group all submitted reports in three categories (100, 90, and 80) depending on how thorough your discussion is. Hope that this last project is more interesting to you. Again, please do not wait until the last moments to do the project. We need to spread out your program testing so that we do not tie down the lab machines. Thanks.

The configuration file

In this project, you are required to implement the program in a way that it should take the specified parameters in configuration file for the sleep time (time between finishing a token and making the next request) and operation time. The name for this configuration file should be ***system.properties***. The format is the following:

```
Multicast.address=239.1.2.3
Multicast.port=54325
ClientNum=5
NumberOfRequests=3
Client1=lin114-01.cise.ufl.edu
Client1.port=49001
Client2= lin114-02.cise.ufl.edu
Client2.port=49002
Client3= lin114-03.cise.ufl.edu
Client3.port=49003
Client4= lin114-04.cise.ufl.edu
Client4.port=49004
Clinet5= lin114-05.cise.ufl.edu
Client5.port=49005
Client1.opTime=500
Client1.sleepTime=1000
Client2.opTime=1000
Client2.sleepTime=2000
Client3.opTime=1500
```

```
Client3.sleepTime=1500
Client4.opTime=1000
Client4.sleepTime=2000
Client5.opTime=500
Client5.sleepTime=2500
```

Output file format

The output format is exactly the same as that in Project 4.

Requirements/Reminder

- Always check and kill all run-away processes.
- Please remember the same documentation requirements are valid for all projects.
- There can be any number of members in the group as specified in the system.properties file.
- The command to start the system must be **java start**
- The name of the configuration file must be **system.properties** and this file should not be submitted.
- The program prints the output to the file, named log.
- Failure to follow the requirements (including the file and program names and submission procedures) will result in points loss.
- Pick random group addresses and port numbers for your tests to make sure you are not receiving others' messages. You may add some unique identifier to the multicast messages (e.g, your login name) and discard any messages, which do not have this identifier.

Additional instructions:

We will use the following scripts to test and execute your project:

```
#!/bin/sh

mv ?*.tar proj5.tar

tar xvf proj5.tar; rm *.class rm *.log

javac *.java; java start
```

Note on Runaway Processes:

Your threads should terminate gracefully. While testing your programs run-away processes might exist. They should be killed (removed) immediately after each testing. The department has a policy on this matter, your access to the department machines might be restricted if you do not clean up these processes.

To check your processes running on a Unix:

```
ps -u <your-username>
```

To kill all Java processes easily in Unix, type:

```
skill java
```

To check runaway processes on remote hosts:

```
ssh <host-name> ps -u <your-username>
```

And to clean:

```
ssh <host-name> skill java
```

Grading Criteria:

Correct Implementation/Graceful Termination:	70%
Report Category	30%
Total	100%

Late Submission Policy:

Late submissions are not encouraged but we rather have you submit a working version than submit a version that is broken or worse not submit at all.

Every late submission will be penalized 15% for each day late for up to a maximum of 5 days from the due date.