

**Summer 2005: CS 438/631/731 Distributed Computing
Homework-2**

200 points. Individual Work Only. Due July 5, 2005 before class.

1. Extend the Chat client-server program implemented in homework-1 to use multiple threads at the server and also provide message logging. Once a new client connects to the server, the server must create a separate thread for communicating with that client. The server must also provide appropriate synchronization when accessing any shared resources such as the list of users in the chat session when users are added or deleted. The server program must also log all messages that it broadcasts in the format: *receivedtime:senttime:message*. Note that the *receivedtime* must correspond to the time the message was received at the server and the *senttime* must correspond to the time the message was sent to each client by the server. Each thread must log the messages in ascending order of the *receivedtime* timestamps and write it to a file named *user.log* where *user* corresponds to the unique userid of the client (passed as argument name). When the same user connects again messages must be appended to this file. The log file is created when the user connects to the chat server for the first time. When the connect method of the server is called the log file is opened and whenever the broadcast method is invoked the timestamps and the messages are written to the file. When the disconnect method called the log file is closed. Note that the server must send the message to the client program that initiated the broadcast request also (unlike homework1). The server must also log any *lookup* calls (of course the server will not be able to log messages exchanged with the gossip method). The client programs must also include the timestamp when requesting the server to broadcast a message. The timestamp must be appended to the start of the message followed by “:” (the message in the broadcast call will have the format “timestamp:message”). The client program must also keep a message log of all messages that it has sent along with the timestamps for the times the message were sent by the client and the time the message was received back from the server (e.g., *senttime:receivedtime:message*).

Write a driver program that will use threads to create multiple instances of the client program and execute different chat clients simultaneously. Use the input redirection to avoid typing in text to each client program. **Execute all programs on the SUN machines (blazer1 through blazer10)**; you can connect to them remotely using a secure shell client. **[180 points]**

2. Test the client and server programs with at least 3 clients and include at least 10 messages per client. Compare and analyze the timestamps in the log files on the client side and server side. You must execute the client and server on two different machines. **[10 points]**
3. Is this version of the program any better than the previous version developed in homework 1 except for the logging feature? **[10 points]**

Hints:

1. Use the method System.currentTimeMillis() in the java.lang package to get local timestamps.
2. Use code segments from Java Almanac to create threads, create files, write files, and close files.

**Summer 2005: CS 438/631/731 Distributed Computing
Homework-2**

For Graduate Students (Bonus question for undergraduate students): [20 points]

Using the timestamps from the client and server programs compute the average time taken to exchange messages between a client and server. Also determine if there is any clock skew between the two systems. If there is clock skew explain how this can be fixed using Lamport's algorithm.

Feedback Questions (answer to these questions has no impact on your grade):

- Was this homework too difficult, or too easy?
- Was the assignment fun or challenging?
- Was there something that was unclear?
- Was the project too long for the given amount of time?
- What did you learn from this homework?

Submission Instructions:

Make two separate directories one for the server and one for the client program:

Following files must be included in the Server directory:

1. Client and Server Service Interface Definitions
2. Server Service Implementations
3. Skeletons for the Server Implementations
4. Stubs for the Server Implementations
5. Stubs for the Client Implementations

Following files must be included in the Client directory:

1. Client and Server Service Interface Definitions
2. Client Service Implementations
3. Skeletons for the Client Implementations
4. Stubs for the Client Implementations
5. Stubs for the Server Implementations

List ALL the references you used in this homework as well as test cases used to test your programs. This includes any classes that you used that you did not write and any help you received from any other sources. Use appropriate class name and include comments to indicate various operations performed by the program. Your program must have the following header information within comments:

```
/*
```

```
    Name:  
    BlazerId:  
    Homework #
```

```
*/
```

**Summer 2005: CS 438/631/731 Distributed Computing
Homework-2**

Create a tar/zip file using the following filename format: blazerId-cs438-hw1.tar or blazerId-cs438-hw1.zip. The tar/zip file must include the source code and class files, txt/word/pdf/ps document for the typed solution to questions 3-4 and the feedback questions, instructions for executing the programs and test data used to test the programs. Do not include any executable files in the tar/zip file, if you include a batch script rename it have a .txt extension. Email the tar/zip file with the subject header "CS 438/631/731 Homework2" to puri@cis.uab.edu. If you do not email with the correct message header your submission will not processed in a timely manner, so please make sure you include the header. Also please do not make multiple submissions. There is no need to turn in any printed solutions for this homework, email submission is sufficient.

Late Submissions:

Submissions must be made on the due date before the beginning of the class. Late submissions will lose 10% for every 24-hour period, up to a maximum of 50% (weekends and holidays count as one 24-hour period). Any submissions made after one-week will receive a score of 0 for this homework.