Part 1 (worth 20%). Short Answer (Answer all 5 questions)

Show all work to receive partial credit. Note: You must do these questions by yourself (NOT with your partner of Part 2). You are required to use a word processing program to write your answers to homework assignments in this class (submissions should be in PDF). Whenever something is unclear in the question, form assumptions, and make those assumptions explicit in your answers. In all answers, be as specific as possible. Limit your answer to each question or each part of a question to a maximum of approximately 1/4 page.

1. Exercise 3.1: A client sends a 200 byte request message to a service, which produces a response containing 5000 bytes. Estimate the total time to complete the request in each of the following cases, with the performance assumptions listed below:
   i) Using connectionless (datagram) communication (for example, UDP)
   ii) Using connection-oriented communication (for example, TCP)
   iii) The server process is in the same machine as the client.

   [Latency per packet (local or remote, incurred on both send and receive): 5 milliseconds
   Connection setup time (TCP only): 5 milliseconds
   Data transfer rate: 10 megabits per second
   MTU: 1000 bytes
   Server request processing time: 2 milliseconds
   Assume that the network is lightly loaded.]

2. Exercise 4.7: Sun XDR marshals data by converting it into a standard big-endian form before transmission. Discuss the advantages and disadvantages of this method when compared with CORBA’s CDR.

3. Exercise 4.20: Discuss whether the following operations are idempotent:
   • Pressing a lift (elevator) request button;
   • Writing data to a file;
   • Appending data to a file.
   Is it a necessary condition for idempotence that the operation should not be associated with any state?

4. Exercise 5.2: Discuss the invocation semantics that can be achieved when the request-reply protocol is implemented over a TCP/IP connection, which guarantees that data is delivered in the order sent, without loss or duplication. Take into account all of the conditions causing a connection to be broken.

5. Exercise 5.12: A client makes remote procedure calls to a server. The client takes 5 milliseconds to compute the arguments for each request, and the server takes 10 milliseconds to process each request. The local operating system processing time for each send or receive operation is 0.5 milliseconds, and the network time to transmit each request or reply message is 3 milliseconds. Marshalling or unmarshalling takes 0.5 milliseconds per message. Calculate the time taken by the client to generate and return from two requests:
   (i) if it is single-threaded, and
   (ii) if it has two threads that can make requests concurrently on a single processor.
   You can ignore context-switching times. Is there a need for asynchronous RPC if client and server processes are threaded?
Part 2 (worth 75%). Programming Assignment (Replicated IM Server)

You are encouraged to do this assignment with one other person from our class. You can do this assignment by yourself if you wish.

Your boss has come to you with the following dilemma: She loves your IM client that you recently developed but is not so thrilled with the (outsourced) IM server. There are two problems with the IM server that she wants you to fix:

1. When the IM server crashes, all of the information is lost. This includes the list of registered users, username/passwords, and the status of each registered client (e.g., logged in, and at which location, etc.)
2. Users get REALLY annoyed when they can’t log in for whatever reason – e.g., the server might be busy, the machine might be down, the server might be down, the network might be down, etc.

Although it would be great if you could just modify the source code for the previous IM server to solve these problems, because of legal disputes with the outsourced company, this source code is unfortunately unavailable. Your boss points out that because this is an open protocol, and because you’re such a wizard at software design and development (after all, you received your degree from the University of Virginia!), she has the utmost confidence that you can write the server from scratch (she points out that you already have a client to test against!)

With regard to the second problem, above, you come up with the following design:

The purpose of the server “frontend” is to be the “outward-facing service” that the clients see (the clients are unaware of the existence of multiple back-end services). This server “frontend” accepts the messages enumerated in the first assignment in the section 2.1 (“Messages to/from the IM Directory Server”). The “server frontend” interacts with the two servers on the right of the figure – essentially if one of them is down, then there’s another one to contact. Note that the purpose of two “backend servers” is not for performance reasons (e.g., load-balancing between the servers) but rather for reliability.

You are required to solve both problems identified above, using this software architecture. The “Server frontend” must open a Java socket, waiting for contact from a potential client. All communication between the server frontend and the “IM Servers” MUST use Java RMI. Reasonable options for persistence include using the file system or using a database system (there are pros and cons for each). It is up to you how many IM servers you use – note that there is a law of diminishing returns here. You are required to support 2 servers, but a “more robust” architecture can earn some measure of extra credit, if you present a reasonable design and justify its existence (in other words, extra credit cannot be earned simply by adding “more”). You determine which information to replicate between the IM servers, and when and how. You also have to design algorithms to run when one of the IM servers goes down (and similarly when one of them comes back on-line). The client programs that you wrote for the first assignment in CS451 MUST run unmodified with this new server system.
To be a little more specific regarding the requirements, you are to be driven by a single use-case: One or more of the replicated servers may be killed, remain off-line for some amount of time, and then be restarted. The interface server won’t be killed, but it’s not allowed to store any of the client data. You must create a design that: [1] maintains service, [2] minimizes client data loss, [3] minimizes message passing.

To get started, here are a couple of good web pages that show the basics of using Java RMI:


It is probably easiest to use your Windows machine, and install Java JDK 6 from [http://java.sun.com/javase/downloads/index.jsp](http://java.sun.com/javase/downloads/index.jsp)

A nice (free) IDE for Java is Jcreator (choose the “LE” edition at [http://www.jcreator.com/download.htm](http://www.jcreator.com/download.htm))

When you want to compile and test, open up a command prompt (note: cygwin is not necessary for this assignment) and first type:

```
path "c:\Program Files\Java\jdk1.6.0\bin";%path%
```

You will then have access to programs such as “rmic” and “javac”.

**Questions to address in your submission**

1. Protocol design: Without discussing the particular implementation choices you made, what is the protocol that you designed to handle the various client messages? What is your approach/protocol before any failures occur? How did you balance the three goals from above?

2. How do you recognize when one of the IM servers goes down? What is your protocol/behavior when one of the servers goes down?

3. Vulnerabilities: What problems can your system handle? What problems can your system not handle? Is a particular window of time in which these problems occur? What are some potential solutions to these problems?

4. What are the pros/cons of your implementation? Would someone actually like to use your system on a daily basis (as compared to Yahoo messenger, for example)? What are some limitations to your implementation?

5. What testing did you do? That is, what did you do to convince yourself that your implementation was correct and/or the performance was sufficient?

6. *[for those with more than 2 servers, for extra credit]* Why did you use more than 2 servers? Can you quantitatively/qualitatively describe/evaluate this design’s value as compared to a 2-server design?

**Grading Policy for this Programming Assignment**

Each program must be properly and reasonably designed, documented, implemented, and tested/evaluated. The grading for this part of the assignment will be:
1. 50%: The quality of the code itself. This includes: the readability of the code, the correctness of the code, the efficiency of the code, etc. (Do not have debugging statements execute in the final version of your program).

2. 50%: The answers to the questions above.

**What to hand in AND email**

The due date for this assignment is 12:30pm on Thursday Mar 1. **Note that the submission requirements are stricter this time, to help us grade:**

- **Partner 1 must**:
  1. Hand in a paper copy of his/her answers to Part 1 (the book questions). This paper copy must clearly state the name of Partner 1, and additionally state “Part 2 done with Partner 2 name”.
  2. Email an electronic copy of his/her answers to humphrey@cs.virginia.edu with the subject line “CS451 Assignment 2 Part 1”.

- **Partner 2 must**:
  1. Hand in a paper copy of his/her answers to Part 1 (the book questions). This paper copy must clearly state the name of Partner 2, and additionally state “Part 2 done with Partner 1 name”.
  2. Email an electronic copy of his/her answers to humphrey@cs.virginia.edu with the subject line “CS451 Assignment 2 Part 1”.
  3. Hand in a paper copy of the team’s Part 2. This paper copy must clearly state the names of both partners.
  4. Email an electronic copy of the team’s Part 2 to humphrey@cs.virginia.edu with the subject line “CS451 Assignment 2 Part 2 (Partner 1 name, Partner 2 name)”.