CS4514 Computer Networks B07

Design Proposal Report 40 Points
Due: Tuesday, November 20, 2007 at 4 p.m.

The final assignment (Program 4) involves the implementation of a "primitive" application on top of an emulated three-layer protocol stack. Each project team must select from one of the three applications presented in program 4.

This assignment is to submit a design proposal in a technically profession style. The paper must be typed and is not to exceed 20 pages.

[Required Team Design Meeting Deadline: Friday November 9, 2007]

After choosing one of the three projects, your first task is to schedule one **required design meeting** with Professor Kinicki to ask questions about the details of the project. The three key points to focus on in your design report are: a **detailed** specification of your proposed application layer; a thorough explanation of the data structures and **communication mechanisms** you propose to use at the layer interfaces (e.g., processes or threads, shared buffers between layers, signals, semaphores) and a **clear**, preliminary discussion of the implementation of data link layer sliding window mechanism.

The Application Layer

Generically, your application layer implements an interactive request/response protocol, and you must support at least **two concurrent clients**. The client abstraction implies reading commands from standard input, handling any auxiliary inputs (e.g., reading in photo files), preparing the appropriate server requests, sending the request as an application message to the server, and printing out the response returned by the Server.

The project team **MUST** propose, design and implement their application layer. Your proposal includes an explanation of the "primitive" functionality of both the client and the server that you will support and the **specification** of your application protocol. You must specify valid commands that the client accepts as input, the meaning of the command request to the server, and the server responses. Your protocol must support at least five distinct commands. Your commands should include at least one large transfer in both directions (e.g., transfer of a photo file).

You are free to design your application layer in terms of the "messages" sent to the network layer. However your design must be able to handle multi-message application layer peer communications such as a file transfer.

If the server encounters an application level error while processing a request, it returns an appropriate error message. Given the limited time to complete this project, you can make 'reasonable' assumptions about what input errors your application interface will handle. Clearly state these assumptions in your proposal..

The Network, Data Link and Physical Layers

Many (but NOT all) of the details for these three layers have been spelled in the program 4 assignment. Your design report needs to specify the missing details that will be in your design.

The **key design decisions** for this project involve your design choices for communication that will dictate the specifics of the interfaces between the layers and the concurrency control mechanisms that you employ (e.g., semaphores, shared memory, pipes, processes and/or threads). Your design proposal must identify your choices.
A Warning: do not get too fancy by using mechanisms that you have not used previously.

Test Data and Live Demo

Since each proposal will be unique, it is each team's responsibility to provide interesting and meaningful test data to demonstrate that your project works. Each project will require a one hour demonstration to show how much of your project works. If the final project turned in does not work completely, then you must provide mechanisms that show which modules work properly. At the live demo, you must turn in a hard copy of your code, your **original** graded design proposal, and an updated addendum and/or a README file that identifies changes in your final design and **clearly states** what works and what is not working.

Output

It is a good idea to implement your project with a debug (or verbose) mode that outputs a very detailed trail of exactly where your clients and server are during execution. However, you want to be able to turn this off for the final demonstration. In conjunction with your debugging, you also need to provide statistics gathering functionality for your concurrent server and all executing clients. These statistic gathering components may or may not be separate processes, but it is quite useful for the demo that these monitoring functions output **ongoing information** into separate windows. The following are a list of "suggested" monitoring statistics that your programs should provide. You may want to add additional monitoring information that is specific to your proposal:

1. the total number of data frames transmitted successfully
2. the total number of data frames received successfully
3. the total number of data frames received with errors
4. the total number of ACK's transmitted successfully
5. the total number of ACK's received successfully
6. the total number of ACK's received with errors
7. the total number of duplicate frames received.

You **must provide** the total execution time for each running client.

Grading of the Design Proposal Report

Each design proposal report will be graded based on basic technical writing standards including: organization and structure, proper grammar, formal presentation style, typos and misspellings, clarity and completeness of the design proposal and clarity of the writing style.

Your report must include: the specification of the application protocol that includes command syntax and parameters, a complete discussion of the concurrency and the layer interface issues, a preliminary discussion of your data link layer sliding window mechanism, a diagram illustrating the components and interfaces of your design, a table that clearly allocates the work among the three team members, a tentative milestone schedule and a list of references or bibliography.

It is critical that a **hardcopy** of your design report is turned in prior to the Thanksgiving break!