CmpE 476 Distributed Systems, Spring 2011, Dr. M. Ufuk Çağlayan
Term Project, Assigned: 18.05.2011 Wednesday, Due: 05.06.2011 Sunday, 16:00 (Final Exam Time)

The purpose of this project is to design a simple Application Layer Protocol (ALP) and implement it by Berkeley sockets programming. ALP should be operational on TCP. Properties of ALP that you will design and implement are as follows.

- ALP is a client-server protocol. There are multiple clients and but a single server.
- ALP is specifically to be called Simple Mp3 Download Protocol (SMDP) and the server is Super Mp3 Server (SMS).
- SMS keeps a number of Mp3 files which are binary files. Also, there is a master file that keeps information about Mp3 files. A typical record of this file will have the unique ID (MID) of an Mp3 file, the title of Mp3 file content (i.e. the name of the song) and the path to the actual Mp3 file.
- Any client can anonymously browse (i.e. display the contents of) the master file to see which Mp3 files exist and also retrieve an Mp3 file. A Mp3 file can be retrieved by its MID number. Retrieval is either by MID number or at random. Client may inquire about the largest MID number in master file. If the supplied number is in the range or request is for a random Mp3 MID, SMS then will supply the Mp3 file and it will be downloaded to the client's machine.
- If you like, you can design and implement the client so that it can request more than one Mp3 files at once by specifying a range for MIDs. In that case, the server will send all the files in that range to the client in a sequential manner. You get additional points if you take this approach.
- Any registered client can add a new Mp3 file to SMS. A registered client is the one who can login to SMS (USER and PASS commands). SMS can also accept new registrations.

All Mp3 files can be stored in the same folder at the server side. I expect you will implement the client/server in C programming language but C++ is also OK (No Java..!). If you use any other programming language, you must justify your selection in the report you will turn in.

Example client server applications that are demonstrating the use of Berkeley sockets will later be distributed. There are a lot of examples on Internet on socket programming anyway.

Note that there are a number of open points in the project and you must make your own assumptions and work alone to finish the project. Team work is not allowed in this project.

You must turn in the following.

- A detailed report of reasonable length, first specifying the problem as you understand it and your further assumptions, then describing the simple application layer protocol that you will design for the client and server to communicate with each other and a discussion of advantages and shortcomings of your design, and then describing the overall design of your application, the environment in which your application can run and your overall discussion. The report must be submitted in paper form.
- Commented source listing (as part of the appendix of your report) detailing your Berkeley sockets implementation with a commentary of what is working and what is not!
- A zipped file, whose name will be **CMPE476-Project-YourLastname**, containing your report, all of your source and binary programs and clear instructions to set up and test your development.
- If everything is OK, you may give a demo of your implementation, if what you have developed is really working. Set up an appointment with the TA.