PROJECT: PHASE#2
CS 4/55231 INTERNET ENGINEERING
Spring 2004, Department of Computer Science, Kent State University
Instructor: Javed I. Khan

Time: <u>3 weeks</u>, Due Date:_____

The objective of the second project is to make our server and client much more HTTP/1.0 like. To do this you need to implement three commands available in HTTP/1.0. HTTP/1.0 supports only three types of commands. In this assignment you will be asked to implement two of them.

4. (500 points) Update your MiniC and MiniS so that if it is started with -v option " can echo all the conversation between the server and the browser on the screen.

RFC1945 contains the specification HTTP/1.0. Your first assignment is to read this RFC (and your class notes), and determine the exact conversation that will continue between a browser and a server if the browser wants to fetch an URL. (a) If the specified URL is invalid. (b) If the specified URL is valid. Assume the URL is the plain HTML file "example.html". Can your MiniC (if you are in client group) or MiniS (if you are in server group) already capable of this conversation? If not make it capable of doing so.

<u>Submission:</u> Copy this version of your program into minis-GP03PROG4.c and minis-GP03PROG4.h files. Make it a part of your master makefile. Name of the executable should be minis-GP03PROG4. If you are in browser group, copy this version of your program into minic-GP03PROG4.c and minic-GP03PROG4.h files. Make it a part of your master makefile. Name of the executable should be minic-GP03PROG4.

5. (1500 points) Using POST command a browser can send document to a Server. Read RFC1945 (and your class notes), and determine the exact conversation that will continue between a browser and a server if the browser wants to POST an URL. (a) If the specified URL is invalid. (b) If the specified URL is valid. Assume the URL is the plain HTML file "example.html". Update your MiniC and MiniS so that if it can perform the POST operation.

<u>Submission:</u> For the server, copy this version of your program into minis-GP03PROG5.c and minis-GP03PROG5.h files. Make it a part of your master makefile. Name of the executable should be minis-GP03PROG5. For the browser, copy this version of your program into minic-GP03PROG5.c and minic-GP03PROG5.h files. Make it a part of your master makefile. Name of the executable should be minic-GP03PROG5.

6. (1500 points) In the server side, in this assignment you will implement a server which can support CGI script , however without any input parameters! In the next few classes, I will explain an example CGI conversation. This assignment is based on that example. In fact you can use the same CGI script. If the GET request (like "GET /cgi-bin/javed/busy HTTP/1.0") arrives where URL points to cgi-bin subdirectory, then assume it is a cgi request. Modify your server so that it can take appropriate action to invoke the executable program "busy". Determine what should be the HTTP/1.0 response if the URL is not there, (b) determine the action if it is there. Modify miniS so that it can support CGI. Use Unix STD/ IO redirection to receive data from CGI script.

Submission: Copy this version of your program into minis-GP03PROG6.c and minis-GP03PROG6.h files. Make it a part of your master makefile. Name of the executable should be minis-GP03PROG6. If you are in browser group, copy this version of your program into minic-GP03PROG5.c and minic-GP03PROG5.h files. Make it a part of your master makefile. Name of the executable should be minic-GP03PROG5.

7. (1500 points) In the client side, In this assignment you will implement a client-pull. Modify your program so that if can read the following tag <UPDATE TIME=ss>. Now modify your program in such a way that, If it finds such a tag inside the document, it waits the specified ss seconds and then again automatically reloads the document.

Submission: Copy this version of your program into minic-GP03PROG6.c and minic-GP03PROG6.h files. Make it a part of your master makefile. Name of the executable should be minic-GP03PROG6.

8. (1000 points) This part will require you to organize your server and client log files so that in each line the server will print the records associated with one and only one incoming request. Now modify your "MINIC" into a Spider. If you invoke it with "– spider log-file-name" It should be able to read the URL's from the log file (server or client log) and be able to send out request for each of them in a sequence. It should however, write new performance logs in its active log.

Submission: Copy this version of your program into minic-GP03PROG7.c and minic-GP03PROG7.h files. Make it a part of your master makefile. Name of the executable should be minic-GP03PROG7.

9. (1000 points) Now take three log files with about 50 URLs listed. The first one should have URL in your server. The second one should have URL's within "kent.edu" domain. The third one should have all URLs outside "kent.edu" domain. Play your MINIC crawler at 4 different times of days. Record all the three sessions from the crawler's log.

10. Submission: Plot a graph showing the retrieval time for each document sets and submit them with your report.

11.

SUBMISSION PROCEDURE

Each group should submit the project codes once. But, the reports should be individual. The project directory should be mailed as a ZIP file by at least one of you. The each of you should individually send your report.

For the Project (50% points):
6. Create an "README2.HTML" file in your "IN2004S" directory. This file should explain:

   6.1. A list of files included in this submission.
   6.2. A clear description how to compile,
   6.3. A clear description that how to use (start, terminate and run) your program.
   6.4. Include any special instruction to the grader if it has unusual interface.

7. Create a ZIP file called GROUP3.ZIP with the "IN2004S" including the README2.HTML file and its sub-directories.

Individual Report (35% points):
8. Each of you should now prepare one document called as "REPOR2-*YOURLASTNAME*.DOC" (It can be HTML, MSWord, or TXT file) as a phase report. For each part of the problem, It should describe:

   1.5. How you have implemented your part of the problem at logical level.
   1.6. If you are using any new API subroutine, you should explain why you are using that.
   1.7. You should also explain the design of your modules. Explain all the user defined variable, key data structure, etc. special files that you have defined to get the job done.
   1.8. One page **Group Member Evaluation Table, with three columns and a row for each member. In the columns describe (a) what part of the project a member has done, (b) your evaluation about the contribution of the member in percentage.**

Code Documentation (15% points):
9. Add documentation In the codes:

   9.1. At the top of each piece of your source code, include the following:
   ```
   /************************************************************************
   GROUP: xxxx
   MEMBERS: 1. Xxx, 2.xxxx, 3.xxxxx etc.
   DATE:
   CS 4/59995 INTERNETENGINEERING  SPRING 2004
   PREPARED FOR PROF. Javed Khan
   ************************************************************************/
   ```

   9.2. In each subroutine inside your source code include the tag:
   ```
   /************************************************************************
   GROUP NAME: xxxx
   MODULE DEVELOPER 1. Xxx, 2.xxxx, 3.xxxxx etc.
   MODULE DESCRIPTION: briefly say what it does? how it does?
   ************************************************************************/
   ```
   9.3. Also include ample comments inside your code explaining the program.

Final Mailing:
10. You now need to mail all of them to the course account:

10.1.　　　Mail the Project ZIP file to IN2004S@maui.cs.kent.edu with subject fields "IN2004S Project 2 GROUP3"

10.2.　　　Mail the Report it to IN2004S@maui.cs.kent.edu with subject fields "IN2004S Project 2 GROUP3: your last name".

Check thoroughly before you submit. Keep a copy of all the files including zip file in your directory. Do not modify them afterward. If need arises, TA may want to check these files. Any modification afterward (reflected in the file date) will result in late submission penalty.

───

Grading:
See notes to grader in the website.

Cheating and Copy:

If a copy is caught, all involved submissions (original as well as the copies) will be penalized. So it is your responsibility to guard your work. Secure the read/write access of your directories. Any copy will result in ZERO grade for the assignment for both parties. Only exception is when you report the theft of your work in advance.