McMaster University
Department of Computing and Software
Dr. W. Kahl

SFWR ENG 3BB4
Assignment 2

SFWR ENG 3BB4 — Software Design III — Concurrent System Design

15 February 2007

This assignment is due on Monday, March 5, at 9:30

First work through chapter 13 in BLP.

(a) Find out what the value of *PIPE_BUF* is on your system, and document how you found this out.

(b) For getting the Chapter 13 CD Database Application (pp. 533ff.) to work, you may need to change the value of *DBM_LIB_FILE* to *ndbm*. It is possible that you need also further changes.

Document all the changes you needed to make to get *make* to produce the server and client programs.

Record these changes as *A2_make.patch*.

(c) The server currently only accepts the command-line flag *–i* to trigger initialisation.

Add the command-line flag *–d* (for "daemon") that lets the server "send itself into the background" after successful startup.

Let the server create a *server.pid* file containing the process number of the (backgrounded) server process, and let the server delete this file when cleaning up for exit.

Test whether the backgrounded server reacts appropriately to signals — document how you perform these tests.

Record the changes for this item (relative to the result of (b) as base) as *A2_bg.patch*.

(d) Add to the server a global "client list" datastructure containing the process numbers of clients that have been communicating with the server, and for each client a time stamp for the last interaction.

Let the clients send a new "quit" message to the server when they are terminating, and let the server delete clients sending these "quit" messages from its "client list".

Make updates to this datastructure visible with *DEBUG_TRACE*.

Record the changes for this item (relative to the result of (c) as base) as *A2_clientlist.patch*.

(e) Let the server check periodically (e.g. every ten seconds) whether any of its clients has been inactive for more than a certain, configurable threshold time, e.g. five minutes, and in that case terminate the client via a signal.

The client should catch that signal and terminate cleanly.

The server should survive this, and you need to document why your solution to this cannot interfere with normal request processing.

Test also the case that the server attempts to terminate a client that has already terminated without sending a "quit" message. Document how you test this, and how the test results show that the server acts appropriately.

Record the changes for this item (relative to the result of (d) as base) as *A2_timeout.patch*.

Deliverables:

- Your **log** on paper or electronically.
- A printout of the patches (context or unified diffs) listed above, each accompanied by documentation of the changes and the underlying motivations and design decisions.
- The patch files, bundled as a .tar.gz file containing one top-level directory, and there at least a REDME file explaining the other files in the package, and how to apply the patches to the textbook code.

Further technicalities of the **submission procedure** will be posted next week on the course page.