SE 3313A – Operating Systems

Design Project: Multi-User Multi-Transaction Server (Due 5:00pm, Dec. 8)

I. Objective

As discussed in class, the project ties together all of the elements we have been using in the first four labs to create a multi-user, multi-transaction server. I expect that this will give you the ability to apply threads, thread synchronization (semaphores, blocks and so on), parallel processing in ways that will give you new appreciation for what they can do.

II. The Big Picture

As before, you will require a lot of information to complete the lab. Accordingly, I have broken the assignment into multiple documents.

The contents of this document explain only what I expect. Additional documents are available that describe the resources available to you.

I would like you to work in **groups of 2-4.**

III. A Multi-User Multi-Transaction Server

When I speak of a "multi-transaction, multi-user" server, I have some parameters in mind, on which I am not flexible:

- The server **MUST** run in the UNIX environment
- Each transaction **MUST** have **AT LEAST** two participants.
- There should in principle be **NO LIMIT** to the number of simultaneous possible transactions.
- Termination **MUST** be **graceful**. That means transactions clean up after themselves in a graceful way. It also means that terminating the server terminates all associated transactions in a graceful way. "**Graceful**", once again, means without core dumps, unhandled exceptions, the need to **kill** the process and so on. However, "graceful" **DOES NOT** mean "polite". A client can be unceremoniously booted out of a dying transaction, as long as no crashes occur.

There are some points on which I am quite flexible:

- The client can run in whatever environment (or set of environments) you like. I'd LIKE it if you used Android clients, but it's not necessary. I will provide a Gameboy-like "skin" for Java and Android "skin" that you can customize if you want.
- The definition of what is a "transaction" is entirely up to you. Could be a game instance, a chat room, some kind of social media thing, whatever.
- The decision on whether **NEW** users can choose to join a transaction already in progress is entirely up to you.

IV. Amazon Web Services

I will provide AWS hosting for your servers. Refer to the document ***Using an Amazon Web Server (includes both PC and Mac instructions)*** in the ***Project Resources*** folder on OWL.

V. Deliverables

I would like the following from you.

- By Monday, November 20: An **email** outlining the composition of your team and the application you plan to create
- By Monday, November 27: Some kind of design document. It can be plain text, as long as it shows how the various parts of your application work together. How many threads do you need? What sort of synchronization objects? How will you handle termination?
- By Monday, November 27: Completion of **ALL Labs 1-4**. The ability to serve multiple sockets, create multi-threaded, properly synchronized Android and UNIX applications are important milestones. I will post solutions to ALL LABS on that day.
- By Friday, December 8: A **demo** of your application. Could be to me or the TAs.
- By Friday, December 8: **Your final submission**. The final report should include an updated design document, your source code and some reflection on lessons learned. Pay careful attention to the issues we have discussed: thread management, synchronization, process termination. Was your original design adequate or did it require modification? Before you ask, yes you can have an extension until the day of the final examination. I think that would not be a wise use of your time, but you can have it if you want it.

VI. Discussion

Finally, as always, I would appreciate your discussion of the course's lab experience as a whole. Was it worthwhile? Did you learn anything? Can you see ways to apply what you have learned? Was it too easy? Too difficult? What ought I to do differently next time? There is not a wrong answer, except no answer at all. If you would like to be critical, that's OK, too, but please use professional language.