CS 494 Homework 2: Due 27 Feb 2006

In this homework, you will complete the first elaboration iteration for the project you described in the first homework. In particular, you will have a fully functioning prototype version of your system ready by the end of this homework.

Changing projects or groups

If you feel that your project is not viable for any reason, or you would rather change projects, you can speak to me about doing so – I don't want to lock you in, for the whole semester, to a project that you will not enjoy.

The same applies for the groups – if you want to change your groups, let me know.

Document Deliverables

You need to include the following documents for this homework.

- **Use cases**: You should provide two more fully dressed use cases. This is probably easiest done after you have done some development, as that will help you figure out what the rest of the use cases will be.
- **Domain model**: As described in the book and during lecture.
- **System Sequence Diagrams**: For the two use cases that you are implementing (see below), you will need to include a SSD for each.
- **Operation Contracts**: You should provide a few operation contracts for the parts of the SSDs that need clarification. Two pages of operation contracts is sufficient.
- **Language choice**: You will need to decide what language you want to use (see below), and write a brief description as to why you chose this language. This does not have to be anything long-winded – a paragraph or two is sufficient.

The documents themselves are going to be on the "short" side – say 3 pages for each of the 2 use cases, and 1 page (in Visio) for each of the 3 diagrams, and 2 pages of operation contracts. The language choice need only be a paragraph or two. Thus, your documents will total about 11 pages.

As before, you will notice that a lot of the finer details have not been laid out. These details are being left to you, as they will vary from project idea to project idea. We are looking for (and will be grading based on) the fact that you have put a lot of thought into this system.

Implementation Language

You may pick any language to implement it in, as long as it fulfills the following requirements:

- It must be an object oriented language, of course
- It must be able to be executed in a Linux/Unix environment (this eliminates C#)
- You must not have had too much experience with the language, and thus do not know it well

The last point will require a bit of explanation. The purpose of this last requirement is to allow you to learn a language that you do not know already. Having written a single, relatively small, program in a language qualifies as not knowing it well. Having taken the language in CS 101 or any successive CS class will generally qualify as knowing it well. For groups, you will need to choose a language that the group members as a whole do not know well. There will be, at some point, something to sign saying that you have followed this rule. As with all the rules for this project, I am willing to bend them if there is a good reason to do so – speak to me about this if it might apply to you.

My expectation is that the language choices will be from the following: Java, C++, PHP, Perl, and Python. If you want to choose a language not listed here, you will need to speak to me first.

Code Deliverables

You will need to implement a working prototype of this system. This doesn't have to be anything fancy – just a run through the main scenario in the two fully dressed use cases from homework 1 is sufficient. If you provided more than 2 fully dressed use cases in homework 1, then you don't have to do all of them. The goal here is to implement maybe 500 or so lines of code *per group member*, as a rough estimate. Most of the development for this project will take place during the following iterations.

All your code will need to be in a code/ subdirectory, as described below. You will also need to provide a readme.odt file that describes what the various files of code do. This doesn't have to be long – just enough so we have an idea of what each file does. If this is clear from the SSD, you are welcome to just state that in the readme.odt file. Lastly, your code needs to be commented so that we have some idea of what is going on. Terse comments are fine, as long as they allow us to understand your code.

Submission

All your deliverables will need to be zipped into a file named hw2.zip, and submitted through the course submission page (http://www.cs.virginia.edu/~cs494/submit.html). The diagrams should be done in Visio, the text documents in OpenOffice. The same formatting rules from homework 1 also apply here (normal margins, normal text size, single spaced, etc.) You can have each of the documents be in separate files, if you would prefer. All of your code should be in a code/ subdirectory in the zip file. And the readme.odt file (described above) should be either in the root directory or the code/ subdirectory. Again, the 5 Mb submission limit is in effect – if your file is larger than that, you will need to let me know.

If your system can not be executed in the normal way (i.e. by compiling and then executing in a Linux/Unix environment), you must provide an alternate means for me to examine and execute the system. This includes any web-based system – you will need to set up the code off of your home page (or other webpage that you choose). This also includes any system that requires external services, such as a database – the easy way to solve this is to provide a means to use a text database instead of a regular relational database. Basically, I need to be able to see your program running if I cannot run it myself. If there are any questions on this, feel free to ask. I can also show you how to set up website execution and/or database access.

The homework is due by the end of the day (11:59:59 p.m.) on Monday, 27 February 2006.