CS 1530 Course Project  
(Part 2, Design)

Assigned: October 6th  
Due: October, 20th, noon.

In this part of the project, you are asked to take your requirements specification and turn it into a system design. You should use UML as your primary design notation and try to use as many design patterns to facilitate your task in this process. You may find the Gang of Four paper that was handed out useful in this respect. For UML, you may find the web site [http://www.uml.org/](http://www.uml.org/) useful – it provides more information about UML than the textbook.

You should write your design document in two parts: (1) high level design, and (2) low level design.

(1) The high level design should focus on the high level design. In particular, it should focus on the classes you use in your design that find a correspondence in the real world. At this level, you only want to define their attributes and methods (without any details on their type signatures). You should also specify the most obvious inheritance relationships that may be present at this level.

(2) The low level design will be a more detailed design. In fact, it may do without some of the classes in (1) because they might turn out to be useful. The low level design will also resolve any ambiguities or imprecision in the relations of the high level design. It will likely introduce new classes that do not correspond to any real world entities but are required to implement the system functionality. You may want to use sequence diagrams to illustrate the interaction and collaboration among the classes to provide the functionality defined in your specification document.

As part of this detailed design, you can get extra credit if you use the design by contract approach for some of your classes by specifying their methods’ pre- and post-conditions and class invariants. Consult pages 233-234 in the textbook for details on how to do this.

**Evaluation criteria:**
Your design will be evaluated using the attached design checklist, which you will also be using to evaluate other group’s designs. Your design should be consistent with your specification document. You are strongly encourage to use UML to write down your design – you can shorten this process by writing down the design patterns instead where applicable, i.e., instead of writing many UML diagrams that represent a design pattern, you can identify the design pattern and how it corresponds to your high level design (specification).

There will be a total of 80 points for your design documents, 30 points for the high level and 50 points for the low level design and up to 20 points for extra credit. A further maximum of 20 points will be earned by evaluating other groups’ work, ten points
maximum for each of the two design documents that you will be given shortly after October 20th.

**What to hand in**

You should hand in your design document describing all your modules, classes etc. The description should be as detailed as necessary and as formal as useful, i.e., try to use UML and design patterns as much as you can – but you are not required to describe everything in UML. We are also not going to pay too much detail on strict conformance to the UML standard, in particular, since it was only touched on in this course. Please send the PDF version of your document to the TA – if you cannot produce PDF, let us know and we’ll generate it from you (e.g., from Latex or Word).
Checklist for Detailed Design Reviews

**Structure**
- Is the pseudocode or other representation format consistent in its level of detail?

**Data**
- Has all the data been properly defined and initialized?
- Is all defined data used?
- Do the data element names and types conform to the project data dictionary?
- Are defaults used, and are they correct?

**Correctness and Completeness**
- Is the detailed design a complete and accurate implementation of the high-level design?
- Are the external specifications of each module complete and testable?
- Have all numerical techniques been analyzed for accuracy?
- Has critical timing been analyzed?
- Has the high-level design memory budget been expanded into further detail and updated?
- Are all functions clearly specified and logically independent?
- Have maintainability issues been adequately addressed?
- Does each module have high internal cohesion?
- Does each module have low external coupling?
- Is the detailed design verifiable?
- Is the logic correct, clear, and complete?
- Have all user interfaces been completely designed?
- Can the termination conditions for loops be realized?
- Can all logic be tested?

**Standards and Traceability**
- Have all detailed design standards been followed?
- Does the calling protocol follow project standards?
- Can all parts of the detailed design be traced back to the high-level design and to requirements?

**Robustness**
- Are error conditions handled in a nondestructive manner?
- Can corrective action be taken by the module that traps an error?
- Are unusual conditions handled reasonably and nondestructively?