### Second Project Deliverable: Software Design (DD1)

Your task for this assignment is to describe each interface or module assigned to your group. Describe for each interface or module:

* the overall behavior of the interface or module (text description)
* its operations (a.k.a. methods, functions, procedures) – give the full signature, including parameters with types and type of return value
* the behavior of each operation (text description)
* its interaction protocol (statechart diagram describing the legal order in which the operations may be invoked)

This information will be entered into design documents. Then you will write code skeletons. All of these (plus the “source code” for any diagrams) will be put in our CM system.

### Detailed Instructions:

**Understand the Design Overview and Architecture.**

Each person reads and understands the Conceptual View and the Design. These documents are posted on d2l (Content/ Project/ Design), and in our CM system (Design folder). Your group should understand the general responsibilities of each module in the system, and the specific responsibilities of your team’s modules.

Negotiate operations of interfaces and modules.

Meet with the appropriate groups to make sure all of your interface(s) and module(s) are satisfactory for the groups that use them, and that all interfaces and modules you need are satisfactory. You may need to set up additional meetings with other groups. Not all team members must attend these meetings, but at least most of the team should attend.

For DD1 you should disregard any references to “Factory” and “Factory method”. These changes will be incorporated for DD2.

Record your design descriptions.

Each layer has its own word document that describes the interfaces of and modules in that layer, named <layer>\_Design.docx. Create the document if yours is the first group to commit your design to the repository. For each interface or module assigned to your team, add a section with:

1. The name of your interface or module.
2. A text description of its behavior.
3. A StateChart diagram describing its interaction protocol. (If the module implements an interface, no new diagram is needed unless the module is providing operations that are not in the interface.)
4. For each operation,
   1. Signature (include parameters with types and type of return value)
   2. A text description of its behavior

Insert your diagrams into the appropriate word document, but put the “source code” for the diagram into the appropriate <layer>\_diagrams folder. You may use StarUML or the violet tool.

**Write Code Skeletons**

Based on your interface or module description, write code skeletons. An interface should have only a .h file, and its contents should conform to our conventions for interfaces. A module should have a .h and .cpp file. The .h files should contain as a minimum all public operations, and the .cpp files should have empty bodies (return 0 if a return value is required). The .cpp files should be compilable.

**Add Your Code Skeletons to our CM System**

1. Using our CM system, check out the directory tree for the entire project.
2. Add your .h and .cpp files to the correct location:
   1. Interfaces (e.g. IX) go in the interface directory of the layer:

Implementation/<Layer name>/interface/IX.h

* 1. A module’s .h file goes in the include directory of the layer:

Implementation/<Layer name>/include/X.h

* 1. A module’s .cpp file goes in the src directory of the layer:

Implementation/<Layer name>/src/<module name>/X.cpp

1. Commit your new files.

### What to turn in

* Deliverable: files you check in (commit) to our CM system (no hardcopy):
  + your design descriptions
    1. update the appropriate word document (including diagrams)
    2. upload your statechart diagram “source code”
  + your code skeletons
* Meeting minutes
* Individual reports