

CS5070 Capstone Syllabus Fall 2009

Instructor: Don Cooley, X72451, M414, e-mail don.cooley@usu.edu; Chad Mano Main 412, X70959, chad.mano@usu.edu; Vicki Allan, Main 429, X72587, vicki.allan@usu.edu

Office hours:

M,T,Th,F 11:00-12:00 (cooley)

M, W 3:30 – 5:00 (Mano)

M,W,F 4:00-5:00 (Allan)

Text: None

Grading: P/F only – A student's grade will be based solely on their project and the professionalism exhibited during the class, i.e. there are no tests or other assignments.

Required class meetings

August 24, 26 7:30 AM, Main 117 Overview and ILM review

September 21, 7:30 AM, Main 117 ILM proposals returned and discussed as needed

Friday, December 11, 7:30-9:20 AM - room(s) to be assigned

This capstone course is an opportunity for students to demonstrate that they have achieved the goals for learning established by the Computer Science Department. The course is designed to assess student learning in a student-centered and student-directed manner which requires the command, analysis, and synthesis of knowledge and skills. The goal of CS5070 is to integrate learning from the courses in your major with the courses from the rest of your academic experience. It requires the application of that learning to develop an Interactive Learning Module (ILM). The specific ILM developed is chosen by the student from a list of ILM's supplied by the instructors. Some of you have not developed web applications before or used specific products such as Java, Subversion or NetBeans. Some of you have not used specific coding standards nor participated in code reviews. Graduates in Computer Science are expected to be able to learn new skills as required by the situation. This capstone experience is meant to simulate expectations found on the job.

While the class is only a single credit of P/F, it is an integral part of the department's overall assessment of student learning. All ILM's will be evaluated by a three faculty member team – Vicki Allan, Chad Mano, and Don Cooley. A student's project will be evaluated carefully in terms of the following criteria:

- ILM Assignment (5 points)
- Technical skills which should be exhibited: (20 points)
 - Quality of coding and programming skills – structure, efficiency, readability, use of appropriate algorithms and techniques, object oriented vs procedural techniques, etc.
 - Adherence to a set of coding standards.
 - Appropriate use of data structures – arrays, trees, queues, linked lists, hashing, etc.
 - Appropriate application and use of software engineering principles

- Understanding of machine architecture and its affect on software
- Operating system and network knowledge - concurrency, scheduling, memory management, interrupt servicing, communications, etc.

For this segment of the project we are interested to determine if you have acquired the skills needed to produce quality code. You will do this by exhibiting in the code for your project appropriate technical skills. For example,

- *A student using a linear search of an unordered list where a binary search of an ordered list is more appropriate is not exhibiting such skills.*
- *Is the user interface well structured, user friendly, and secure?*
- *Is the code “bug free”?*
- *Is the code written so that it is readable and extensible?*
- *Were your choices of which algorithm to use technically appropriate, e.g. Quicksort vs bubble sort, recursion vs iteration, etc.?*
- *Is the code efficient in terms of execution time and memory utilization?*
- *etc.*

While a given project need not exhibit all of the technical skills listed above, it should have sufficient technical content to be a significant project.

- **Communication skills which should be exhibited:**
 - **Written communication via the following kinds of documents and software: (10 points)**
 - Requirements Specification
 - Project Plan
 - Detailed Design Documentation
 - Testing Plan and Results
 - User Guide
 - Well written user interface
 - **Oral Presentation ~10 minutes excluding questions (10 points)**
 - Clarity and Completeness
 - Delivery quality

For this segment of the project we are interested to determine if you have a sufficiently clear understanding of software engineering principles and possess the required communication (written) skills so that you can provide quality documentation and successfully implement and test your project. Unlike many class programming assignments we are interested to see that you can:

- *Thoroughly define your project before beginning to code*
- *Develop a plan for the project with appropriate time lines and schedules*
- *If done in a team, appropriately allocate and schedule project responsibilities. There should be documentation in the project proposal write-up to clearly show the responsibilities of each team member. Remember, each team member should have the opportunity to exhibit the kinds of technical skills we are interested in reviewing.*

- **Professionalism skills exhibited**
 - **Setting of work schedules and the meeting of those schedules**

As part of the class, students will be expected to meet deadlines set for the class, i.e. turn in selected elements of the project on time. Please note and adhere carefully to the assigned timeline and work schedule. There is no latitude given if a deadline is not met.

In order to pass the class, i.e. receive a P grade, a student must accumulate at least 40 points out of a total possible of 65 points.

Timelines and work schedules for the class

1. 8/24 Introductory meeting, syllabus review, and discussion of class goals.
2. 8/26 How to implement and integrate an ILM into CSILM environment
3. 9/3 Turn in “test” ILM (5 points)
4. 9/14 – E-mail your ILM write-up through eagle using the **ILM Proposal Form** describing the ILM you plan to develop. It is due by 5:00 PM (5 points). In addition to the specific CS concept(s) to be taught, and the general means in which you will guide students in learning the concept, the write-up should include a description of what technical skills will be included in the project. If done by a team (of up to two members), identify team members and the division of work. Responsibilities should be divided so that each team member has the opportunity to exhibit a significant portion of the required technical and communications skills, e.g. one member should not be assigned documentation as their sole responsibility. Use the ILM proposal form included on the class web site at (digital.cs.usu.edu/~cooley/). It is appropriate to discuss with a faculty member your proposed project prior to this deadline.
5. 9/21 – Review of ILM proposals will be completed and returned to class members. Included in the review will be a technical quality (TQ) factor of 0.0 to 1.0. The TQ will be a multiplier for the points you receive at the end of the semester. Thus, if your total score at the end of the semester is 50 points, and your TQ was 0.8, then your final score would be 40 points. If you feel your TQ is too low, then you may want to “expand” your proposed project to raise this factor. For teams, a single TQ will be applied to all members. Each student will also be assigned a faculty supervisor for the project (Allan, Cooley, or Mano)10/5 - Project plan, requirements specification, and testing plan submitted through eagle. It is due by 5:00 PM (5 points).
6. On-going meetings as needed with faculty supervisor.
7. 11/30 Code Review Meet during class to critique code of other students. Come prepared with four written copies of the code you have produced for the course.
8. 12/4 - CD containing all source code, and written documentation turned in. If changes have been made to the requirements specification, please note them your documentation. Be sure to include a discussion of your testing results. Turn in the CD directly to the faculty mentor assigned to your ILM. Due by 5:00 PM (10 points, -6 points if this deadline is not met)
9. 12/11, 7:30-9:20 AM. Each student (or team) will be given 20 minutes (maximum) during the scheduled final time for the class to give their presentation. The presentations will be given in multiple rooms to be assigned, depending on the number of presentations. Be sure to come early to pre-load any materials needed for your presentation.

Notes:

- A team may include no more than 2 members.

Definitions:

Requirements Specification: Elicitation, definition and documentation of **what** (*not how*) the software is supposed to do. The specification can be conveniently divided into functional and non-functional specifications. (User) interfaces should receive a proper attention. A good specification should be correct, unambiguous, complete, consistent, verifiable, modifiable, traceable and ranked by priority. Unified Modeling Language (UML) is a good tool to use (use cases and use case diagrams in particular). Use also data flow diagrams (DFD) and/or entity-relationship diagrams (ERD) wherever applicable.

Project Plan: Definition of measurable goals; estimation of work effort, schedule and resources; allocations of people, process and facilities; and identification of project risks. You can use work breakdown structure (WBS) and/or a Gantt chart.

Detailed Design Documentation: Functional decomposition of the requirements. Use UML (class diagrams in particular). Use also sequence diagrams and state diagrams wherever applicable.

Testing Plan and Results: Verification and validation plan of the software. Address testing plan at different stages: unit testing, functional testing, integration and system testing, and acceptance testing. Depending on the project, configuration testing, performance testing and stress testing are applicable. Specify test criteria and test cases considering the “Lack of continuity” problem in empirical testing.

User Guide: Specify the prerequisite of the software system. List known issues. Show the steps of a good representative program run session with examples and screen shots.

Interactive Learning Module: A computer-based instructional tool that teaches a particular computer science concept. In addition to teaching the concept, it serves as a self-assessment tool for the student.