CSC 650: Capstone Project (I)
Syllabus

Dr. Chenglie Hu  Phone: 524-7170  e-mail: chu@carrollu.edu

CSC650 is offered every fall, spring, and summer. Using emails is the preferred communication means between students and the instructor, although appointments can be made to meet in person if necessary or desirable.

This class is the first of two-course sequence that serves as the Capstone for the Masters of Software Engineering. In this course students will:

- Select a project (preferred from the student's place of employment if relevant)
- Research the project for existing references, solution strategies and techniques
- Choose a software development process (Unified Software Development Process, or an agile process) that you are going to follow through
- Implement the first two phases of the UP – Inception and Elaboration
- Documents to be created include (but not limited to) ones for software requirement specification, software analysis and design, risk management, and software project management (IEEE compliant, a template is available at: http://www.mastering-project-management.com/support-files/project-management-plan-template.pdf)

Tentative Schedule

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Date</th>
<th>Informal Meetings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground rules and discussions on the logistics of the course</td>
<td>Week One</td>
<td>between students and instructor</td>
</tr>
<tr>
<td>Status report on the Inception phase</td>
<td>a mid-semester week</td>
<td>between students and instructor</td>
</tr>
<tr>
<td>Status report on the Elaboration phases</td>
<td>a week toward the end of semester</td>
<td>between students and instructor</td>
</tr>
<tr>
<td>Final Presentation</td>
<td>end of semester</td>
<td>open to the public</td>
</tr>
</tbody>
</table>

This course CSC650, together with CSC651, is meant to measure whether the student has mastered the material to meet the following goals of the Masters of Software Engineering Program.

Carroll College MSE Program Outcomes

Upon completion of the MSE program, the student will be able to

1. Apply sound software engineering principles and methodologies in any software development process regardless of roles they may play as software developers, development leads, or software project managers.
2. Problem-solve (mostly for business problems) at a higher level using enterprise resources, major Web software development frameworks, and sound software design methodologies.
3. Be competitive in making sound judgment on any IT issues that are related to software development.
4. Meet challenges of a software development process as information technologies advance.
5. Be self-motivated and highly effective players in any team environment.

**CSC650 Course Objectives/Outcomes**

Upon completion of the course, the student will
- Demonstrate the ability to understand a business problem and create a solution with appropriate technology (MSE Goals 1 - 4)
- Demonstrate the ability of executing the Unified Software/Agile Development Process (MSE Outcomes 1 - 5)
- Demonstrate the ability of requirement and risk management (MSE Goals 1 - 5)
- Demonstrate the ability of software analysis and understanding of object-orientation (MSE Goal 1 - 4)
- Demonstrate the ability of creating extendable software design (MSE Goals 1 - 4)
- Demonstrate the ability to learn the latest development technologies and frameworks (MSE Goals 1 - 4)

**GRADING**

This class will be graded by the CSC Faculty and based upon the following criteria.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>The quality of the partially completed product or solid prototypes, documentation produced in the first two UP phases, and a testing plan [MSE Goals 1 - 4]</td>
<td>90%</td>
</tr>
<tr>
<td>The quality of status report, final presentations and regular communication with the instructor [MSE Goals 1 - 4]</td>
<td>10%</td>
</tr>
</tbody>
</table>

- A – perfect or near perfect in all areas
- AB – a major deficiency in one of the key areas or minor problems in a few areas, but perfect or near perfect in other areas
- B – good but not perfect in most areas; major deficiency is identified in one area or minor problems are identified in multiple areas
- BC – major deficiencies are identified in multiple areas, but the work put in was adequate
- C – bad quality across most areas or inadequate progress, but the work appeared adequate
- D – bad quality across most areas, work is inadequate, or little progress demonstrated
- F – bad quality across the board with little evidence of adequate work, or virtually no progress
References

There are more references online than you can possibly read just about all aspects of software development, design, implementation, and testing. So browse as you need. For a reference for writing in Computer Science, read at:
http://www.csus.edu/wac/WAC/Students/Major%20Sites/computer_science.html.

DELIVERABLES (what's in your binder)
The following deliverables are expected from each student.

- All the documentation produced in the first two UP phases
  - Software Requirement Specification document (which typically includes the current set of software system requirements, feasibility study, business modeling, etc.).
  - Software Project Management Plan

Note that for those who have followed company's practice in developing software specs and planning, you include such relevant documents instead.

- Activity report that clearly indicates
  - Iterations performed (that is, the tasks performed and duration of each iteration)
  - The interaction with clients if appropriate
  - Work journals/logs with time you spent on activities (and a total number of hours worked).
- Software Analysis and Design artifacts
  - Architectural design in terms of 4+1 architectural views; these views would be constructed through use cases and dynamic modeling diagrams (activity, sequence, state, dataflow); data modeling with ER diagrams; some detailed design (and you may verify the design with pseudo-codes)
  - Note that in your SRS document, you can simply indicate the fact that the analysis and design artifacts are available elsewhere)
- Test plan (unit, regression, and integration), and the tests that have already been performed thus far if any
- Describe the prototypes developed, and whether they have played a role in the requirement gathering (you can include the screen captures of the prototypes or partially finished product)
- A brief reflection of CSC650 to get you ready for CSC651
- A CD that contains the electronic versions of all the documents

Final Presentation

The final presentation is a formal presentation where the student presents not only the partially finished project but also the software development process he or she has followed along.
The Analysis and Design Documents

The final design documents are part of the final report and should be consistent with the completed portion of the project. These will be based upon the 9 UML categories of documents, although not all may be used on every project. It is expected that the student will use their experience to properly determine how many and when they are used. For example, if you are going to use a true object-oriented development framework, such as .NET or J2EE, to implement your project, the use-case and state/activity diagrams and/or sequence diagrams are normally expected as part of analysis workflow, whereas the detailed class diagrams, ER diagrams, client-objects diagrams, and algorithmic design are normally expected as part of design workflow. The use of design patterns, Hibernate framework, .NET3.5 Entity Framework, distributed components, or any other latest technology (such as AJAX) is highly recommended as appropriate. It is highly possible that your design will be re-factored in CSC651 as you move forward to develop the rest of the software system.

1. Contact Walter Young Center if you need accommodation due to disabilities or other health issues.
2. The instructor and the University reserve the right to modify, amend, or change the syllabus (course requirements, grading policy, etc.) as the curriculum and/or program require(s).
3. ACADEMIC HONESTY: If you significantly use some existing documents or code (not created by yourself), make sure you make an appropriate acknowledgement. For more information refer to the "Academic Dishonesty" policy in VII of the College Student Handbook.