CS 2750
Object-Oriented Analysis and Design (with the Unified Modeling Language)
Summer 2003 Syllabus

Instructor: Delroy A. Brinkerhoff
Office Hours: 12:00 - 1:00 TTh and following the night class
Office Location: TE 110B
Phone: 626-7345
Web Page: http://icarus.weber.edu/home/dab/ (please see “CS 2750” under the “Classes” section)
E-Mail: dab@cs.weber.edu (Please do not use dbrinkerhoff@weber.edu or send HTML encoded email)

Time and Room: 9:45 - 12:00 TTh TE 105

Texts: The Object-Oriented Approach by John W. Satzinger and Tore U. Ørvik
Teach Yourself UML in 24 Hours, Second Edition by Joseph Schmuller

Students with Disabilities: Any student requiring accommodations or services due to a disability must contact Services for Students with Disabilities (SSD) in room 181 of the Student Services Center. SSD can also arrange to provide course materials (including this syllabus) in alternative formats if necessary.

Prerequisites: CS 1220

Grading: Assignments 20%
Quizzes 50% (approximately every other week, lowest score dropped)
Class Presentation 20%
Presentation Attendance 10% (you must be on time)

<table>
<thead>
<tr>
<th>Grade</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>73.3%</td>
</tr>
<tr>
<td>A-</td>
<td>70.0%</td>
</tr>
<tr>
<td>B+</td>
<td>66.7%</td>
</tr>
<tr>
<td>B</td>
<td>63.3%</td>
</tr>
<tr>
<td>B-</td>
<td>60.0%</td>
</tr>
<tr>
<td>C+</td>
<td>total &lt; 76.7%</td>
</tr>
<tr>
<td>C</td>
<td>total &lt; 73.3%</td>
</tr>
<tr>
<td>D</td>
<td>total &lt; 70%</td>
</tr>
<tr>
<td>D-</td>
<td>total &lt; 66.7%</td>
</tr>
<tr>
<td>E</td>
<td>total &lt; 60%</td>
</tr>
</tbody>
</table>

Quizzes: Most quizzes will be scheduled in advance. A few quizzes will be unannounced. As long as students are prepared for and participate in class, there will be little need for unannounced quizzes, on the other hand. . . . The lowest quiz score will be discarded.

Assignments: Assignments will be posted on the class web site listed above. Assignments may not be submitted by email.

Attendance Policy: CS 2750 is a very conceptual class. The main goal of CS 2750 is the development of an understanding of abstract analysis, design, and modeling techniques. The major learning activities are reading the texts, formal classroom presentations (which will occasionally include material not found in either text), and classroom dialog. It is imperative that you read the texts prior to class, that you attend class. It is assumed that all students have organized schedules that permit to do this and that they have budgeted sufficient time (typically 12-16 hours per week). This may necessitate making arrangements with employers regarding travel, meetings, training etc. or coordinating vacations and other special events with family and friends. Students are responsible for material covered during absences regardless of the reason for the absence. Punctuality is expected (habitual tardiness is not acceptable) and attentiveness (not reading the newspaper, surfing the web, working on other classes, or sleeping) is appreciated.
Bad Weather Policy: Please do not take unnecessary risks in inclement weather. The attendance and quiz policies will be relaxed in the event of bad weather. Also, if there is a possibility that class will be canceled due to weather, listen to the radio and watch the class web site above: I post there as soon as I know the class status.

Class Procedures: We will typically meet as a class in the classroom each TWTh. Questions about the reading, the assignments, or the labs are welcomed at this time. Demonstrations, presentations, and class discussions on the assigned reading topics will follow. We will meet in the lab on Mondays to work on assignments.

Schedule: Reading assignments and copies of the presentation slides will be available on the class web page prior to each class. Please visit this page at least weekly.

Disclaimer: This syllabus is subject to change at any time. Alterations made in class supercede this document. Please see the web page for current information.

Course Description: Object-Oriented Analysis and Design, the title of CS 2750, is a subset of software engineering. Analysis is concerned with understanding some part of the “real world” (called the problem domain) sufficiently well to permit writing a program or building hardware to solve some problem within that domain. Object-Oriented Analysis (OOA) is the application of the Object Model to analysis. The goal of OOA is to express a problem domain in the form of collaborating objects and to then generalize those objects into classes. Design is concerned with the expression of the analysis results in an implementable form (either software or hardware). Object-Oriented Design (OOD) adds detail to the problem domain classes discovered during analysis and adds the implementation classes necessary to build the final solution. The Unified Modeling Language (UML) is the current best-of-class methodology for performing OOA and OOD and is our vehicle this semester for learning Object-Oriented Analysis and Design.

Course Goals: To understand software engineering models and life cycles.
To understand the object model.
To understand analysis and design.
To understand how to expresses analysis and design in terms of the object model.
To learn the basics of the UML.
To understand OOA, OOD, and the UML within a software engineering context.
To prepare each student for advanced object-oriented programming.

Objectives and Outcomes: At the conclusion of the course you will:
1. Be able to explain the uses and advantages of engineering models
2. Be able to describe and to use the object model
   a. Describe the difference between a class and an object
   b. Define attributes and behaviors
   c. Represent a class and all if its features with a UML diagram
   d. Illustrate the relationships between classes graphically
   e. Given a class diagram, convert the diagram, including attributes, behaviors, and relationships into C++, Java, or another object-oriented programming language
3. Be able to recognize, to describe, and to use main features of the UML
   a. Major Elements, Rules, and Common Mechanisms
   b. Major Elements: Things, Relationships, and Diagrams
   c. Things: Class, Interface, Collaboration, Use Case, Active Class, Component, Node, Interaction, State Machine, Package, and Note
   d. Relationships: Inheritance, Realization, Association, Aggregation, Composition, and Dependency
   e. Diagrams: Class, Object, Component, Deployment, Use Case, Sequence, Collaboration, Statechart, and Activity
4. Be able to identify, to describe, and to use other software engineering techniques and tools