Fall 2006

CEG 460/660: Introduction to Software Computer Engineering

John A. Reisner
Wright State University - Main Campus, john.reisner@wright.edu

CEG 460/660: Introduction to Software Engineering

Fall Quarter, 2006

Course Description

This course introduces established practices for engineering large-scale software systems. Emphasis is placed on both the technical and managerial aspects of software engineering, and the software development process. This includes techniques for requirements elicitation, analysis, design, testing, and project management. The course emphasizes object-oriented development with the Unified Modeling Language (UML). Hands-on experience is provided through individual homework problems and a partnered project.

Textbook

T.C. Lethbridge and R. Laganiere, *Object-Oriented Software Engineering, Second Edition*, McGraw Hill, 2005. This is a required textbook for this course.

Reading Assignments

Each week of lectures has one or two corresponding reading assignments from the course textbook. Some students might want to accomplish this reading prior to the lecture; others may prefer reading the text after the lecture. Still others might prefer to skim the reading assignment prior to the lecture, with more in-depth reading taking place after the lecture is finished. Students are encouraged to figure out which method works best with their particular style of learning. Whichever method the student decides to use to incorporate assigned readings into the learning process, it should be remembered that the lectures and readings are intended to supplement each other, not act in lieu of each other.

References Available for Student Use

Students may find the following documents helpful:

- *The Rational Unified Process, Version 2002.05.00.25*, IBM Corp, 2003. Available at L:\eng students\csce593\RationalUnifiedProcess\index.html.
- *OMG Unified Modeling Language Specification, Version 1.5*, Object Management Group, 2003. Available at L:\eng students\csce593\Papers\UML 1.5 Spec 03-03-01.pdf.

Instructor Contact Info

John Reisner
Office Hours by Appointment
Daytime Phone: 255-3636 x7422 (this is a WPAFB phone number)
email: **john.reisner@wright.edu** (it wouldn't hurt to cc: **john.reisner@afit.edu**)
→ or use WebCT email tool

The instructor is an adjunct faculty member. Most contact will be done via WebCT, or in after-class discussions. Other meetings can be arranged.

Course Objectives

Each student should be able to

- Comprehend the advantages of using a sound software engineering methodology for developing complex software systems.
- Comprehend the importance of a good software process for managing the development of large software systems.
- Analyze the characteristics of software process models
- Comprehend how the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) can be used to measure and improve an organization's software development process.
- Apply object-oriented techniques and modeling with the UML to the development of software systems.
- Identify and apply other appropriate techniques and tools for use at each phase of the software lifecycle.

Grading

15% Homework Assignments

- Homework assignments are designed to facilitate deeper comprehension about a lecture topic (in other words, these are "think and respond" assignments).
- There may be up to two assignments per week, but some weeks may have one or zero assignments.
- Answers to these homework assignments generally run about half page to one page in length, and should not take too long to complete.
- Details about these assignments will be found on WebCT.
- Normally, these assignments will be due on Tuesday each week (one week to complete a Monday assignment; five days to complete a Thursday assignment). Any exceptions to this policy will be mentioned when the homework is assigned.
- Assignments are due at the start of the class/lab session; please have them printed out and ready to turn in at the start of class. If you are unable to attend class, email will be accepted. Emailed assignments should be timestamped before class time (skipping class does not give you a homework extension).
- These assignments will be graded using the SUE grading system (explained on the following page).

35% Group Project

- Partnered with another student, you will perform a use-case driven design and refinement project.
- This project will begin during the first week of class, and last for the duration of the course.
- Students will submit their work at various stages of development in order to get some feedback.
- Grade for the project is based on the final turn-in. Modifications can be made to interim work before the final grade is assigned.

25% Mid-term Exam

- Mixed-format exam, administered in class.

25% Final Exam

- Comprehensive, mixed-format exam, administered on the last day of classes.

Final course grades will be assigned at the instructor's discretion, after all work has been graded, and the grade distribution has been analyzed.

Class Project

The class project schedule is as follows:

Project Milestone	Initially Due
Partnerships formed, project descriptions	After Week 1 (i.e., one week after first class)
First 5 (or 6)* use-case scenarios	After Week 2
Remaining 4 (or 6)* use-case scenarios	After Week 3
Use-case diagram	After Week 4
Initial class diagram	After Week 5
First two sequence diagrams	After Week 6
Next four sequence diagrams	After Week 7
Remaining sequence diagrams	After Week 8
Final Report	After Week 9
Supporting Documentation	Evaluated in Final Report (Nov 9)

*Project teams with only 460 students require 9 scenarios. Project teams with 660 students require 12 scenarios.

NOTE: Although materials are initially due on the dates indicated, your final project grade is based on the quality of the materials on the day of the final turn-in. Just as in the real world, projects that start off poorly, with problems identified early in the life cycle, will have the opportunity to be rectified. Factors for the final grade include quality, consistency, and completeness of the design, organization and readability of the documentation, and apposite leveraging of object-oriented concepts. More information and guidance concerning the class project will be discussed in class and distributed in various handouts.

The SUE Grading System

Homework for this class will be ordinarily graded on a three-tier scale: the work will be graded as Satisfactory, Unsatisfactory, or Exemplary.

If your submission was Satisfactory, then your grade will be S, which will translate to a 90. Don't think of a 90 as "losing 10 points;" think of it as getting ample credit for satisfactory work.

Occasionally, I receive an assignment with great originality and insight, reflecting much forethought and effort. Not only do I find these assignments enjoyable to read, I sometimes find myself thinking, "This is as good as or better than anything I could put in an answer key." Such exemplary work is graded E, which translates to 100. If submitted work indicates either a lack of understanding of basic concepts, or an apparent apathetic carelessness, then it will be graded as Unsatisfactory, and a numeric grade will be assigned accordingly. If I think the problem lies with misunderstanding the basic ideas, then I will usually provide some personal feedback, with the aim of helping you understand the material better.

Of course, I also reserve the right to stray from this guideline some. For example, I might grade with values such as 85 or 95 (corresponding to S- or S+), or even an S++, which would be a 98. After reading 25 or so essays the same topic, I get a pretty good idea of which papers are more well-thought-out than others. The ones that are "more than satisfactory" receive grades such as 93, 95, or 98, while the truly superior works will receive an E (100). Again, don't ask me what was wrong if your grade is a 90. A 90 means you understood the assignment and did a good job of presenting your response.

I also reserve the right to deduct points for late assignments, depending upon how late the work was turned in, how much advanced notice I was given about when I could expect the work, and any extenuating circumstances that may have applied.

Overall, my goal is to assign homework that requires thought, thereby reinforcing understanding and increasing retention.

Course Schedule (possibly subject to change)

PART I. USE-CASE DRIVEN OO DEVELOPMENT			
Week/Lesson	**Date**	**Lesson Topics**	**Reading Assignment**
1 / 1	Tue Sep _5_ (1st class day)	Course Introduction Overview of O-O Software Development	• Chapter 1
1 / 2	Thu Sep _7_	Objects and Classes Encapsulation Inheritance	• Chapter 2
2 / 3	Tue Sep _12_	Use-Case Scenarios Use-Case Diagrams	• Chapter 4
2 / 4	Thu Sep _14_	Structural Modeling Candidate Classes CRC Cards Entity, Boundary, Control Objects	• Chapter 5
3 / 5	Tue Sep _19_	Behavioral Modeling Sequence Diagrams Polymorphism	• Chapter 8
PART II. THE SOFTWARE ENGINEERING LIFECYCLE			
3 / 6	Thu Sep _21_	Lifecycle Models Overview of Project Management	
4 / 7	Tue Sep _26_	Requirements Analysis	
4 / 8	Thu Sep _28_	Software Design	• Chapter 9
5 / 9	Tue Oct _3_	Implementation	
5 /10	Thu Oct _5_	Testing Types of Testing	• Chapter 10
6 / -	Tue Oct _10_	NO LECTURE MIDTERM EXAM	
6 / 11	Thu Oct _12_	Software Maintenance	
7 / 12	Tue Oct _17_	TBD	
PART III. SOFTWARE ENGINEERING MANAGEMENT			
Class	**Date**	**Lesson Topics**	**Reading Assignment**
7 / 13	Thu Oct _19_	Software Project Management Work Breakdown Structures	• Chapter 11
8 / 14	Tue Oct _24_	Development Processes The CMM	
8 / 15	Thu Oct _26	Software Metrics	
9 / 16	Tue Oct _31_	Agile Development Methodologies Extreme Programming	• Chapter 12
9 / 17	Thu Nov _2_	Configuration Management Risk Management Ethics	
10 / -	Tue Nov _7_	TBD	
10 / -	Thu Nov _9_	TBD	