CS 4560/CS5560 Software Design & Development I
Fall 2012

Instructor:

Dr. Chang Liu
Email: liuc @ ohio . edu
Office: 321C Stocker Center
Phone: (740)593-1249

Lectures: TT 4:35-5:55pm, ARC102

Office Hours: TT 9:00-10:30am

Objectives:

The purpose of the course is to provide students with knowledge and skills needed in the software engineering profession. This course does not focus exclusively on the *implementation phase* of software development, which is only one step in the modern software development processes employed by software engineering professionals. The skills taught in this course involve all the phases of a typical software development life cycle.

Another goal of the course is to provide students with knowledge and experience relevant to working in a software engineering team and interacting with customers. Therefore, the course covers material about interpersonal communications, both with members of a team and with customers. Students apply a software development process while working in teams to engineer a software product for a customer. In addition, the course aims to teach students skills to quickly evaluate and master new technologies.

Students who complete CS4560/5560 learn how to

- perform all major phases of the software engineering lifecycle (requirements, analysis, design, implementation and testing),
- use the Unified Modeling Language (UML),
- perform unit, integration, and system testing,
- effectively participate in software engineering teams,
- lead software engineering teams,
- resolve conflicts,
- interact successfully with customers,
- make formal presentations of software engineering products, and
- evaluate and learn new technologies.

To accomplish these objectives, this course adopts active learning approaches and involves substantial writing. In addition, to facilitate teamwork and after-class interactions among team members, cutting-edge, innovative collaboration and communication tools may be used during and after classes and during office hours.

CS 4560 Course Outcomes

- B: Ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
- B1: Student is able to decompose and map user requirements to specific, measurable and achievable computing requirements.
- B2: Student is able to determine the time and space to meet the computing requirements.
- B3: Student is able to determine the performance of various solutions.
- C. Ability to design, implement, and evaluate a computer-based system, process, component or program to meet desired needs.

- C1. Student is able to implement a software module and demonstrate that it compiles and links without errors.
- C2. Student is able to design and implement a software module and justify that it meets stated requirements.
- C3. Student is able to develop a test harness for a provided software module and use it to call and test the module.
- D: Ability to function effectively in teams to accomplish a common goal.
- D1. Student demonstrates the ability to develop complex team projects in a timely fashion by managing meetings effectively, recognizing principles of constructive conflict management, being accountable for individual tasks, and giving and receiving constructive criticism.
- D2. Student understands the importance of team dynamics by identifying pros and cons of different team structures and selecting a structure most suitable for a certain situation.
- E: Understanding of PROFESSIONAL, ETHICAL, LEGAL, SECURITY and SOCIAL issues and responsibilities.
- E1. Student is able to demonstrate an understanding of issues dealing with intellectual property, including software licensing.
- E2. Student is able to demonstrate an understanding of privacy issues, including HIPA.
- E3. Student is able to demonstrate an awareness of security threats to computers and computer networks.
- E4. Student is able to recognize the importance of professional responsibility in decisions made by informational professionals.
- F: Ability to COMMUNICATE effectively with a range of audiences.
- F1. Student is able to demonstrate appropriate use of language and appropriate word choice for the audience during a technical oral presentation.
- F2. Student is able to demonstrate an organizational pattern that is logical and conveys completeness in a written work that argues a position.
- F3. Student is able to apply the rules of standard English.
- G: Ability to analyze the local and GLOBAL IMPACT of computing on individuals, organizations and society.
- H: Recognition of the need for and an ability to engage in CONTINUING PROFESSIONAL DEVELOPMENT.
- H3: Student is able to recognize post-graduate educational opportunities (short-courses, seminars, graduate school, etc.)
- I: Ability to use CURRENT TECHNIQUES, SKILLS, and TOOLS necessary for computing practices.
- I2. Student demonstrates ability in using standard software design techniques (e.g., UML class diagrams, separation of implementation from interface).
- I3. Student demonstrates ability to use testing techniques to verify that a computer-based system satisfies a given set of requirements (e.g., unit testing, regression testing).
- K: Ability to apply design and development principles in the construction of software systems of varying complexity.
- K1. Student understands the software life cycle by explaining how software design, development, and deployment take place in phases.
- K2. Student understands the major software design and development principles by explaining the concepts of "divide-and-conquer" and abstractions.
- K3. Student demonstrates the ability to apply software design and development principles in large, complex projects by successfully completing term projects.
- K4. Student understands the importance of documentation by providing up-to-date technical documents along with his or her project submissions.

Textbook:
Bernd Bruegge and Allen H. Dutoit, "Object-Oriented Software Engineering: Using UML, Patterns and Java," 3rd edition, Prentice Hall, 2009.

Schedule:
(subject to change)

Week	**Topics**
1/2	Course introduction Introduction to the software engineering process Introduction to interpersonal communication in software engineering Technology evaluation
3	Project requirements
4/5	Requirements capture
6	Analysis
7	Design
8/9	Implementation
10/11/12	Testing
13	Data flow testing
14	Project presentations and demonstrations

Course Grade Components:
- 60% - in-class activities, homework assignments, quizzes, and exams
- 5% - software engineering project log.
- 35% - software engineering project artifacts, including project presentations and demonstrations.

An individual team member' grade may be higher or lower than the team grade, depending on his/her project participation and contribution as determined by team members.

Grading Policy:

Letter Grade	Score (%)
A	90-100
B	80-89
C	70-79
D	60-69
F	0-59

All issues regarding grades must be raised within a week after grades are posted. After one week, all grades are frozen and cannot be changed. Note that CS5560 students will be required to complete additional assignments. In addition, they will be expected to demonstrate strong leadership abilities in teamwork. Their grades are interpreted differently.

Late Policies: 1-48 hours after the deadline - 20% penalty; 48 hours to one week after the deadline – 50% penalty; after one week – 100% penalty.

Bonus Points: Bonus points will be given to students who win awards, provide service to the class, or do exceptionally well in homework assignments.

Penalty Points: Some quizzes, in-class activities, and homework assignments are not graded. However, students may receive penalty points if they do not complete these assignments professionally. In addition, students may receive penalty points if they miss classes, have low peer evaluation scores, or do not complete surveys in time. In addition, the following actions may result in penalty points:
- Failure to use informative log messages in GIT/SVN/CVS.
- Checking in damaged files to GIT/SVN/CVS repository without verification.

- Failure to subscribe to required class/team mailing lists. (If any student has any valid reason for not subscribing to any mailing list, inform the instructor at the beginning of the class.)

Attendance and Makeup Policies: Attendance and participation in all class sessions is mandatory. Class sessions may involve activities in which students will learn the course material. Some of the in-class activities will be graded. Many class sessions will include a quiz to assess comprehension of the material covered in the readings and activities from the previous class. Since in-class activities usually require use of the textbooks, you are required to bring your textbooks to each class meeting.

Makeup exams/quizzes/in-class activities, and project deadline extensions will not be granted, except for legitimate reasons (see student handbook for details [http://www.ohiou.edu/studentaffairs/handbook/]).

Academic Misconduct: Individuals performing plagiarism, copying and other forms of academic misconduct (see student handbook for further details) will receive an 'F' in the course and referral to Ohio University Judiciaries.

Note: Copies of student work may be included in the Teaching Portfolio of Dr. Liu. The work may be viewed by other faculty for the purpose of assessment of the course or educational research, and may be used as examples in future classes. If you do not wish to have your work included in the portfolio, please notify Dr. Liu.

Depending on the project hosting site used in the class, client requests, and other factors, all student work may be released under the MIT open source license, the Berkeley open source license, or other open source licenses. The code produced may be used by the public. Future students may further improve on your work. If you and/or your team do not wish to release your work under these licenses to the public, please talk to the instructor and make special arrangements.

In-Class Activities
To learn the course material, students perform activities during most class sessions. (Thus, this is not a lecture-based course.) The in-class activities will include reflective writing, applying course material to case studies, and small-group discussion. The products of some of the in-class activities will be graded. When grading the assignments, the focus will be on how seriously the task was taken. A grade of *satisfactory* (2 points) will be given when a serious effort was made to address the topic of the assignment; otherwise, a grade of *unsatisfactory* (1 point) will be given.

Software Engineering Team Project
One of the fundamental concepts taught in this course is a software engineering process. The reason for this is that even moderately mature software development organizations employ a documented, standardized process. In addition to applying a rigorous software process, a software engineer needs to function productively in a team. To obtain experience in a team project, teams of students apply the software engineering process and group communication skills taught in the course to solve a real problem. Students also learn how to make a formal presentation to a customer and have the opportunity to practice this skill through presentation of the project artifacts. Each team will make presentations of the software products that it produces. *Each student* must make part of each presentation.

This syllabus is subject to change at the discretion of the instructor. Changes will be announced in class.