1 Course Description

This course explores the software development process. It will discuss the major activities common to software development processes, and some of the ways in which those activities are organized and managed.

Heavy emphasis will be placed on the day-to-day skills required during software construction. The course will explore lessons and tools offered by the major successful open-source software efforts.

The course requires each student to participate as a member of a team in a significant team project. Each student will be required to demonstrate proficiency in several software development tools.

1.1 Instructor

Thomas Kennedy
Dragas 1100H
(757) 683-7725
tkennedy@cs.odu.edu
Please make sure to include the course name “CS350” in the subject line of any email related to this course.

1.1.1 Office Hours

Office hours are posted online at http://www.cs.odu.edu/~tkennedy/

1.2 Course Pre-requisites

- CS 330 (Object-Oriented Programming and Design), or
- CS 361 (Advanced Data Structures and Algorithms)
  - Students who have not taken CS 330 are encouraged to take CS 382 (Introduction to Java) as a pre-requisite or, at the very least, to work through that courses’s website during the first few weeks of the semester.

1.3 Meeting Times and Delivery Method

Students must register for both a lecture section and a recitation section.

- Lectures: No regularly scheduled meetings

  Lecture materials for this course will be available on-line, primarily in printed/text form. A small number of class meetings may be conducted by network conferencing. Most notably, this will include an orientation session held during the first week of the semester. Attendance at these will be optional and recordings will be made available for those unable to attend.


  Recitations will be conducted by network conferencing.
    - The primary purpose of the recitation is to allow project teams to meet with the instructor for periodic evaluation of their progress.
      - For the purpose of these meetings, it will be important that students be seated at or connected to their chosen development machine and prepared to demonstrate aspects of the project under review.
      - There will also be a small number of earlier recitations devoted to learning the conferencing tools that will be used in the later meetings.

Recitation sections will not meet every week, but the scheduled meetings are mandatory. During weeks when recitations are not scheduled, project teams may wish to use that time for their own meetings.

As necessary, instructor may announce open “help” sessions during the recitation time periods to provide aid on assignments and labs. Attendance at these sessions will be optional.

2 Basic Course Information

2.1 Required Text

Readings from the internet will be assigned from the course website.
2.2 Resources for Getting Started

- ODU Online Student Orientation

2.3 Computer Accounts

Students will need two network accounts to participate in this class:

- An ODU ITS account. This is the account associated with your @odu.edu email. It will allow you to log into the course’s Blackboard site. All ODU students automatically receive this account, though you may need to activate yours, if you are new to ODU.

- An account on the CS Dept. network. This will be used for access to the CS dept computing resources, and for accessing and submitting assignments. You may have a CS account already if you were registered for a CS class last semester. If not, the account setup and password can be initiated at http://www.cs.odu.edu/ by clicking on “Account Creation” under “Online Services”.

2.4 Hardware & Software Requirements

Students will need frequent access to a PC capable of hosting software development activities or of connecting remotely to CS Dept servers where such activities can be performed.

Students will be attending network conferences requiring the use of a microphone. Webcams are optional.

For both remote access to CS Dept servers and for network conferencing, a good-quality internet connection is important.

The course will introduce students to a wide variety of software packages. All of these are open-source, free software, but students will need to install some of these on their chosen development machine (whether their own PC or in their account on the CS network).

3 Course Policies

3.1 Attendance

Project review sessions will be scheduled for selected weeks during the recitation periods. Attendance at these is mandatory. Failure to attend will result in substantial grade penalties for that portion of the project.

3.2 Due Dates and Late Submissions

Late assignments and make-up exams will not normally be permitted.

Exceptions to this and other grading policies will be made only in situations of unusual and unforeseeable circumstances beyond the student’s control, and such arrangements must be made prior to the due date in any situations where the conflict is foreseeable.

“I’ve fallen behind and can’t catch up”, “I’m having a busier semester than I expected”, or “I registered for too many classes this semester” are not grounds for an extension.
3.3 Academic Honesty

Everything turned in for grading in this course must be your own work, or, for team projects, the work of your own team. Opportunities for teamwork will be clearly identified as such.

Students are expected to conform to academic standards in avoiding plagiarism.

The instructor reserves the right to question a student orally or in writing and to use his evaluation of the student’s understanding of the assignment and of the submitted solution as evidence of cheating.

Students who contribute to violations by sharing their code/designs with others may be subject to the same penalties. Students are expected to use standard Unix protection mechanisms (chmod) to keep their assignments from being read by their classmates. Failure to do so will result in grade penalties, at the very least.

This policy is not intended to prevent students from providing legitimate assistance to one another. Students are encouraged to seek/provide one another aid in learning to use the operating system, in issues pertaining to the programming language, or to general issues relating to the course subject matter.

Student discussions should avoid, however, explicit discussion of approaches to solving a particular programming assignment, and under no circumstances should students show one another their code for an ongoing assignment, nor discuss such code in detail.

Violations of this policy will be reported to the Office of Student Conduct and Academic Integrity for consideration for punitive action.

3.4 General University Policies

The ODU Catalog lays out a wide variety of University policies that are binding upon both students and faculty. All students are required to abide by these.

3.5 Grading

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>15%</td>
</tr>
<tr>
<td>Semester project</td>
<td>45%</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>15%</td>
</tr>
<tr>
<td>Final exam</td>
<td>25%</td>
</tr>
</tbody>
</table>

Grading is normalized.

4 Topics

Topics will include:

- Software development processes, including the waterfall, unified OO, and extreme programming models.
- Revision management: local, centralized, & distributed approaches
• Configuration Management: project configuration, managing external libraries
• Documentation tools
• Build Management
• Test-driven development
• Unit & Integration Testing: coverage, self-checking, mocking, designing for testability
• Debugging: local & remote, monitors, reverse debugging
• Regression Testing
• Issue tracking
• Software Forges & Repositories

4.1 Objectives

Students completing this course should be able to:

• Demonstrate an understanding of the overall strategy of software development:
  ○ Discuss the phases and component activities of software development
  ○ Assess the likely impact of popular software process development models on a project.
  ○ Discuss common team organizations and roles in software development.

• Work with software requirements documents
  ○ Read common forms of requirements documents
  ○ Write at least one standard form of requirements document
  ○ Apply requirements to guide the subsequent construction of software

• Apply best practices in collaborative software construction
  ○ Discuss the issues and problems involved in collaborative development of software.
  ○ Evaluate the suitability of alternative best practices for a software construction project.
  ○ Support common best practices of via a modern IDE and associated tools
  ○ Apply a variety of software measurement and estimation techniques.

4.2 Expectations

Students will engage in team projects in this course. Students are expected to actively participate in and contribute to their teams, and this engagement with the team will be part of the grade.

© 2015-2016, Old Dominion Univ.