

# Syllabus

## CS 350: Introduction to Software Engineering

### Fall 2005

**Prerequisites:** CS 361 or 330. You must have completed at least one of these with a C or better before taking CS 350.

**Meeting Times:** Lecture, Tues., Thurs.. 8:45—10:00 a.m., on-campus site: Gornto 201

Recitations:

Teletechnet:

Tues. 10:15-11:05

On-campus:

Friday, 9:00--9:50. Hughes 1110

Friday, 10:00—10:50 p.m. Hughes 1110

Note that recitations do not meet the first week.

**Instructor:** Dr. C. Michael Overstreet  
E&CS 3206  
Phone: 757-683-4545  
web: <http://www.cs.odu.edu/~cmo>  
e-mail: [cmo@cs.odu.edu](mailto:cmo@cs.odu.edu)

**Office Hours:** 3:00—4:15 p.m. Tues. Thurs. Other hours by appointment.

**Texts:** *PSP: A Self-Improvement Process for Software Engineers*, Watts S. Humphrey, Addison-Wesley, ISBN 0-321-30549-3, 2005.  
*Introduction to the Team Software Process*, Watts S. Humphrey, Addison-Wesley, ISBN 0-201-47719-X, 2000.

#### **Course Description:**

Software is not developed by individuals working independently, but by teams. For most projects, the development of many individual components by separate groups must be planned and coordinated. Central requirements for doing this are: 1) the ability to predict accurately the time and effort required for each component, and 2) each developer's ability to deliver high quality, reliable, documented components on time. Many now believe that the key for organizations to be able to develop quality software on time and within budget is the developers' use of an effective software process. Central to the success of any process is the abilities of the software developers who define, develop, and validate the software product.

This class will focus on two areas. The first is on a process you can use to improve your personal software development skills with a particular focus on three areas: 1) refining your software development ability through the use of a software development approach similar to that used by many commercial developers, 2) improving your ability to determine the time required to complete a software development task, and 3) enhancing your ability to deliver high quality software components on time. The second general emphasis area is team projects: how to organize, plan, design, and work in teams. Effective teams need people with strong technical skills (for example, designing, programming, testing, configuration management), with good person skills (working with both other team members and with customers, coordinating tasks with other project teams, coordinating people within the team).

This course is not about programming; we assume you already can do that. The course is about the process used to develop quality software and the skills needed to work in groups.

This course has a heavy emphasis on metrics. Some are personal: (for example, prediction accuracy, productivity, and quality); others are team related (for example, conformance to schedules). While you will compute some numbers as measures of your own performance, your numbers will not be compared with anyone else and your grade in no way is affected by how your numbers might compare with other class

members. What is important is that you learn how to compute these numbers and how to use them to improve your own skills. The data you turn in with each assignment will be graded for completeness, accuracy (at least consistency), but the values (e.g., whether design took 5 minutes or 5 hours) will have no effect on your grade. We will provide aggregate class data so that you can see how your numbers compare with other class members, but your data is your personal information. We will only use the data to ensure that you are following the required process, that you know how to collect the data, and so that we can compute overall class statistics (so that you have something to compare your data with).

Course benefits to you: Upon successful completion of this course, you should have improved:

- 1) your ability to predict how long a software project will take,
- 2) your software development productivity (more debugged lines of code per hour of effort), and
- 3) the quality of your software (fewer errors per 1000 lines of code).
- 4) your ability to organize new project teams and to play a key role in the success of the team.

Programming assignments will be evaluated normally: Correct execution matters, as does providing appropriate documentation in the specified style, structure, design, and provision for reusability. Your goal should be generating correct, reusable code. For more details see the on-line 350 material.

Support material (handouts, forms, schedules, assignments, examples, readings) for this class is located at [~www.cs.odu.edu/~cmo](http://www.cs.odu.edu/~cmo); follow the links to CS350. This location will be updated throughout the semester with additional material.

This is a project-oriented course and the student is advised that projects will be time consuming.

**Recitations:** Unlike some other computer science classes, recitation attendance in CS 350 is required. Assistance with project assignments will be provided. Some additional material will be presented in recitations (particularly on use of software development tools). Project details will also be discussed and you will have an opportunity to ask questions about projects and other class material. In the second half of the semester, the part of the recitation period will also be used for group meetings.

**Projects & Grading:** Five individual programming assignments and one team project will be assigned.

Exams will be in-class, closed book, covering class notes, text reading assignments and outside reading assignments. The final exam is comprehensive.

**Honor Code:** The honor code applies to all project components and examinations; while verbal discussion among individual class members is encouraged, any work turned in for a grade should be the work of the person turning the component in for credit. Design, test data and code sharing is a violation of the honor code.

**Students with Special Needs:** If you have special needs (e.g., visual, mobility-related), you should let the instructor know so that appropriate accommodations can be made.

**Grading:**

Individual projects:	20%
Team project:	20%
In-class exams:	25%
Recitation assignments, quizzes:	10%
Final Exam	25%

**Project Teams:** Several assigned tasks are team-based. The instructor will assign teams (with input from you). So your grade will be impacted by how your team performs. Your team will be responsible, for example, for creating test plans, test data, and conducting testing using class data (you will test other team members' code). However, your project grade is based primarily on the quality of your own components (e.g., code, documentation, test plans, test data).

**Lateness Policy:** This class is project oriented. Success is possible only if you keep up. For this reason, if you turn in an assignment up to one day late (from one minute to 24 hours late), a 10% penalty will be assessed, up to three days late, a 20% penalty will be assessed, one week late, a 30% penalty will be assessed. After this, no credit will be given unless special arrangements were made with the instructor beforehand. Many people in the class have full-time jobs; some actually have personal lives. I will work with you to handle emergencies if I know beforehand and I can be convinced that you can still complete all course objectives by the end of the semester. If you have problems meeting deadlines or with other course requirements, talk to me as early as possible. Being informed after-the-fact does not generate sympathy, particularly in a class this size.

Assignments are due by midnight on the date specified.