16.355 - Advanced Software Engineering

Nancy G. Leveson
Fall 2001

Instructor: Prof. Nancy Leveson
Room: 33-313
Tel: (617) 258-0505
Email: leveson@mit.edu

Objective: By the end of the class, you will be able to evaluate software engineering techniques and approaches. You will also be able to exercise professional judgment in selecting an approach for a particular project based on an understanding of how the present state of software engineering practice came about, what was tried in the past, what worked and what did not work, and why.

Required Background: This class is an advanced class and as such will not provide basic instruction and practice in programming or in specific approaches or tools for developing software. An introductory class in software engineering is the prerequisite, but this prerequisite can be waived if you are willing to read a basic software engineering textbook prior to the class or as the class proceeds. You will gain the most from the class if you have previously participated in some software engineering team project and have experience writing a substantial program, but this experience is not a prerequisite.

Reading: There is no textbook for the class although it will be helpful to have a general software engineering textbook available to look up specific topics that you need more background on than you already have. There are lots of these available and any one will do (all are pretty much the same).

Instead of a textbook, required reading will consist of a carefully selected set of historically important and foundational papers as well as more current ones reflecting the latest thoughts. Some papers will be technical while others will be opinions or essays. The literature is vast, and papers have been selected for their historical relevance in the development of the field or for their ability to help you critique the assumptions underlying current software engineering dogma.
Assignments: There will be no programming assignments. Because my goal is to enhance your ability to make successful professional judgments rather than use a specific set of tools or a particular approach to engineering software, most assignments will involve evaluation and interpretation rather than practice in applying particular techniques or tools. A few assignments will involve actually trying out a technique (such as doing a small proof of correctness) where such application is important for understanding the basic concepts or for comparing approaches.

There will be two types of assignments: Before each class session, you will be required to submit a short (one half to one page) summary of each paper assigned for that week along with your answers to a few specific questions on that topic or paper. Because class sessions will involve a lot of discussion of the readings, late summaries will not be accepted. In addition, there will be a few assignments involving applying some technique or gathering information about actual practice in an industrial context.

There will be no exams aside from the last week’s summaries, which will involve synthesizing what you have learned in the class.

Class Sessions: Class meetings will include a mixture of lecture and discussion. You will find that most concepts are easily learned from the readings and do not require my explaining them to you. A few topics are difficult to learn on your own or I could not find any good tutorial papers on them so they will be described in class.

Grading: Grades will be based on the weekly assignments.

Schedule: Software engineering is a vast topic that cannot be covered in one semester. The specific topics to be covered have been chosen somewhat arbitrarily and may change in future offerings of the class. A schedule and list of assigned readings will be posted.