Course Syllabus

CS 490 - Software Engineering

Dr. Steven K. Andrianoff

Fall, 2017

Time and Place

Lecture: TTh 10:00 - 11:15 Walsh 103
Lab: To Be Announced

Office Hours: 1:30 – 2:30 Mon
1:30 – 2:30 Tues
2:00 – 3:30 Wed
2:00 – 3:30 Thur
11:30 – 12:30 Fri
Other times by appointment
*(times subject to change during first week)*

Office: Walsh 113
Phone: 375-2053
E-mail: andrianoff@sbu.edu

Catalog Description:

The course provides an overview of software requirements analysis, the software design process, verification and validation, software maintenance, and documentation. A major component of the course is a project that provides experience in the analysis and design of a software product using an object-oriented methodology. Prerequisite: Computer Science major with senior level status. 3 credits. Fall.

Further Course Description:

The student studies the nature of the program development task when many people, many modules, many versions, or many years are involved in designing, developing, and maintaining a software system. The primary emphasis of the course is the introduction and application of a specific object-oriented methodology for the specification, design, and implementation of a software system. An overview of relevant software engineering issues such as project planning, requirements analysis, software maintenance, and documentation will be included. A major component of the course consists of working in a team on the design of a software system. The object-oriented methodology will be applied to this project and many software engineering issues will be addressed in the context of this project.

Texts:


Note: Additional readings from various articles and books will be assigned as well.
Topics:

1. Object-oriented software development using the Larman text
   (Including the following process and modeling technologies)
   - Iterative development (chapters 2, 3)
   - Requirements (chapters 4, 5, 7)
   - Use Case modeling (chapter 6)
   - System sequence diagrams (chapter 10)
   - Domain modeling via UML (chapters 9, 11)
   - Interaction diagrams (chapter 15)
   - UML class diagrams (chapter 16)
   - GRASP principles (chapters 17, 18, 25)
   - GoF design patterns (chapter 26)
   - UML state machine diagrams (chapter 29)

2. Additional software engineering issues:
   - Models of software evolution
   - Software project planning
   - CASE tools
   - Configuration management

Course Policies and Grading:

Grades will be based on:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quizzes</td>
<td>20%</td>
</tr>
<tr>
<td>Homework</td>
<td>10%</td>
</tr>
<tr>
<td>Reaction papers</td>
<td>10%</td>
</tr>
<tr>
<td>Term paper</td>
<td>10%</td>
</tr>
<tr>
<td>Software project</td>
<td>50%</td>
</tr>
</tbody>
</table>

Quizzes -
Approximately five quizzes will be given that cover the material presented in lectures.

Homework -
Homework assignments will consist of exercises based on the lecture material and assigned readings.

Reaction papers –
One reading assignment from the Brooks text or an additional source will be assigned each week along with a one-page reaction paper based on the reading.

Term paper –
A 4-6 page paper on the ethical/moral/societal aspects of software engineering will be assigned in November.

Software project -
One-half of the course will involve applying an object-oriented methodology to the analysis and design of a software product. The class will be divided into teams that each work on a separate project, developing a set of deliverables (documents). Lab time outside of class will be devoted to the project. The lecture portion of the course will cover the terminology, concepts, and software engineering techniques to be applied in the project. The final exam for the course will be a class presentation of the project. All students on the team will participate in the presentation. See Lab Overview for more details.
Attendance policy:

Attendance at the lectures is expected. Attendance at the labs is mandatory. Attendance will be monitored. I consider more than two absences from the lectures to be excessive.

Classroom Etiquette

Use of cell phones or other personal electronic devices during class is rude and inconsiderate of others. Please turn off cell phones and similar devices during classes and labs. This policy does not apply to personal laptops which may be used for notes and lab exercises.

Academic Honesty Policy

The Computer Science Department takes academic honesty violations (e.g., plagiarism) seriously. The department issues a pamphlet, Academic Practices and Policies, that outlines our expectations in this area. All assignments, including work done both inside and outside of the lab, are governed by this document. Inappropriate conduct and the penalties for it are defined in this document. A copy is distributed to all computer science students each semester. Please review it: Ignorance of the policies and procedures is not an excuse for violating them.

Academic dishonesty in any form will not be tolerated. Typically the first offense will result in a zero on the assignment. Repeated offenses will likely result in a failing grade for the course. Any offense deemed punishable will also be referred to the Dean of Arts and Sciences.

Students with Disabilities

Students with disabilities who believe that they may need accommodations in this class are encouraged to contact the Disability Support Services Office, Doyle Room 26, at 375-2065 as soon as possible to better ensure that such accommodations are implemented in a timely fashion.
Computer Science Goals and Learning Objectives

Software Engineering is the capstone course for the Computer Science program. The course addresses the following goals and objectives that have been established by the department:

Goal 1: Discipline Specific Learning
Students will be able to understand and apply the theoretical tools of computer science to standard problems from the field.
Objectives:

4. Students will describe the design principles used in the construction of particular software systems.

Goal 3: Communication Skills
Students will be able gather requirements for a system from third parties, choose a paradigm in which to design the solution, and communicate the parameters of that solution to both professionals in the field and the originating parties.
Objectives:

2. Students will simulate the process of software design as it would be carried out within a professional company.
4. Students will be given the opportunity to present results of their work in oral and written forms; this will include the presentation of posters and/or papers intramurally and extramurally.