CS 410: Introduction to Software Engineering

1. Course number and name: CS 410: Introduction to Software Engineering

2. Credits and contact hours: 3.000 Credit Hours, 3.000 Lecture hours

3. Instructor or course coordinator: Stan Kurkovsky


   a. other supplemental materials
      - Course syllabus and lecture notes are available on line at: http://www.cs.ccsu.edu/~stan/

5. Specific course information

   a. brief description of the content of the course (Catalog Description):
      An examination of the software development process from the initial requirement analysis to the operation and maintenance of the final system. The scope of the course includes the organization of software development projects, the verification and validation of systems, the problems of security and privacy, and the legal aspects of software development, including software protection and software liability.

   b. prerequisites or co-requisites: CS 253

   c. indicate whether a required, elective, or selected elective course in the program: Elective

6. Specific goals for the course

   a. specific outcomes of instruction, ex. The student will be able to explain the significance of current research about a particular topic.

   Program objectives and outcomes are supported by the following learning outcomes achieved by students upon a successful completion of this course:

   - Be able to identify, formulate, and solve software engineering problems, including the specification, design, implementation, and testing of software systems that meet specification, performance, maintenance and quality requirements;
   - Be able to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of a software development project;
   - Be able to function effectively as a team member;
• Understanding professional, ethical and social responsibility of a software engineer;
• Participate in design, development, deployment and maintenance of a medium scale software development project;
• Be able to convey technical material through oral presentation and interaction with an audience;
• Be able to convey technical material through written reports which satisfy accepted standards for writing style;
• Be able to use Unified Modeling Language in software specification documents;
• Be able to evaluate the impact of potential solutions to software engineering problems in a global society, using the knowledge of contemporary issues and emerging software engineering trends, models, tools, and techniques.

b. explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course.

The course outcomes will allow students taking CS 410 to make progress toward the following department outcomes:

• Outcome (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline.
• Outcome (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
• Outcome (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.
• Outcome (d): An ability to function effectively on teams to accomplish a common goal.
• Outcome (e): An understanding of professional, ethical, and social responsibilities.
• Outcome (f): An ability to communicate effectively.
• Outcome (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues.
• Outcome (i): An ability to use current techniques, skills, and tools necessary for computing practice.
• Outcome (j): An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
• Outcome (k): An ability to apply design and development principles in the construction of software systems of varying complexity.
7. Brief list of topics covered

- Software processes
- Agile software development
- Requirements engineering
- System modeling
- Architectural design
- Design and implementation
- Software testing
- Software evolution
- Socio-technical systems
- Dependability and security
- Project management
- Project planning

8. Expected Performance Criteria

Students working in small teams will participate in a semester-long course project aimed to apply in practice many of the topics covered in this course. Throughout the semester, students will produce a number of tangible deliverables corresponding to the milestones within an iterative software development lifecycle: project proposal, system requirements, system architecture design, iterative development plan, iterative development reports, software deployment document, project demonstrations, etc. Student understanding of software engineering principles will be demonstrated by their performance on two tests and a final exam.