Basic Course Information: CS 580

I. Catalog Description
Software metrics and models. Software development methodologies. Advanced topics in object-oriented software engineering. Formal methods for modeling and specification. Software architecture. Software testing. Survey of trending software engineering tools and researches.

II. Required Coursework and Background
Pre-requisite(s): CS 480 or consent of instructor.

III. Expected Outcomes
On successful completion of this course, students will be able to:

1. Apply software engineering theory, principles, tools and processes, as well as the theory and principles of computer science and mathematics, to the development and maintenance of complex, scalable software systems.
2. Design and experiment with software prototypes
3. Select and use software metrics
4. Participate productively on software project teams involving students from a variety of disciplines
5. Communicate effectively through oral and written reports, and software documentation
6. Elicit, analyze and specify software requirements through a productive working relationship with project stakeholders
7. Evaluate the business and impact of potential solutions to software engineering problems in a global society, using their knowledge of contemporary issues
8. Explain the impact of globalization on computing and software engineering
9. Interact professionally with colleagues or clients located abroad and overcome challenges that arise from geographic distance, cultural differences, and multiple languages in the context of computing and software engineering
10. Recognize the need for, and engage in, lifelong learning and researching
11. Demonstrate software engineering application domain knowledge and research ability

Outcomes of this course will build student capacity in each of the following areas as defined by programmatic objectives for the computer science major.

P-SLO 3. An ability to build applications, either individually or in a team, that are robust, reliable, and maintainable.

P-SLO 4. A breadth of advanced knowledge and skills in applied areas of computer science.

IV. Instructional Materials
Text:
[GoF95] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.
[Martin08] Robert Martin, Clean Code: a Handbook of Agile Software Craftsmanship, Pearson Education, 2008.

References:
[Brooks86] Fred P. Brooks. (1986). "No Silver Bullet — Essence and Accident in Software Engineering". Proceedings of the IFIP Tenth World Computing Conference: 1069-1076.
[Gelperin88] David Gelperin and Bill Hetzel, "The Growth of Software Testing," Communications of the ACM, vol. 31, no. 6, June 1988, pp. 687-695.
[Mens04] Tom Mens and Tom Tourwe, "A Survey of Software Refactoring," IEEE Transactions on Software Engineering, vol. 30, no. 2, February 2004, pp. 126-139.

[Parnas72] David L. Parnas. (1972). "On the Criteria to be Used in Decomposing Systems into Modules". Commun. ACM 15, 12 (December 1972), 1053-1058.
[Parnas75] David L. Parnas and Daniel P. Siewiorek. (1975). "Use of the Concept of Transparency in the Design of Hierarchically Structured Systems". Commun. ACM 18, 7 (July 1975), 401-408.
[Zhu97] Hong Zhu, Patrick Hall, and John May, "Software Unit Test Coverage and Adequacy," ACM Computing Surveys, vol. 29, no. 4, December 1997, pp. 367-427.

V. Minimum Student Material
Textbook and class handouts

VI. Minimum College Facilities
A classroom with a projection system, a computer laboratory

VII. Course Outline
1. Introduction to software engineering, process models & methodologies
2. Agile development, SCRUM
3. Software project planning & estimation
4. Requirements capture, analysis & modeling
5. Advanced software design & architecture
6. Version control, build automation, and continuous deployment
7. Aspect-oriented Programming
8. Clean code and code refactoring
9. Emerging software development trends
10. Professional software development workflow and best practices
11. Software scalability and cloud computing
12. Software maintenance
13. Software metrics

VIII. Instructional Methods
Lecture
Problem-solving/Discussion
In-class exercises
Small group activities
Project-based learning

IX. Evaluation of Outcomes
A. Student Assessment
    1. Homework assignments
    2. Team projects
    3. Midterm exam
    4. Final exam
    5. Rubric

B. Meaningful Writing Assignment
Short answer essay questions on exams will require students to explain and justify their response in writing.

C. A Matrix of Course Student Learning Outcomes vs Methods of Assessment
If the course is being evaluated for accreditation purposes, approved department accreditation assessment tools will additionally be utilized.

Course Learning Outcomes	Methods of Assessment				
	Homework assignments	Team Project	Midterm Exam	Final Exam	Rubric
Apply software engineering theory, principles, tools and processes, as well as the theory and principles of computer science and mathematics, to the development and maintenance of complex, scalable software systems.	x	x	x	x	
Design and experiment with software prototypes	x	x	x	x	
Select and use software metrics	x	x	x	x	
Participate productively on software project teams involving students from a variety of disciplines	x	x	x	x	
Communicate effectively through oral and written reports, and software documentation	x	x	x	x	
Elicit, analyze and specify software requirements through a productive working relationship with project stakeholders	x	x	x	x	
Evaluate the business and impact of potential solutions to software engineering problems in a global society, using their knowledge of contemporary issues	x	x	x	x	
Explain the impact of globalization on computing and software engineering	x	x			x
Interact professionally with colleagues or clients located abroad and overcome challenges that arise from geographic distance, cultural differences, and multiple languages in the context of computing and software engineering	x	x	x	x	x
Recognize the need for, and engage in, lifelong learning and researching	x	x			x
Demonstrate software engineering application domain knowledge and research ability	x	x			x