Department of Computer Science

College of Science and Mathematics
Montclair State University, Richardson Hall

MONTCLAIR
STATE
UNIVERSITY

Spring 2015

I. **Course**: CSIT 415-01 & 02 Software Engineering II

Instructor: Dr. Hubert A. Johnson

Office: W314; Ph 973- 655-7238
Office Hrs. M 3:00- 3:50 pm;
T 4:00 pm - 4:50 pm.
R 12-12:50 pm

Email: johnsonh@mail.montclair.edu

Text: Software Engineering, Theory and Practice 4th . ed., Shari L. Pfleeger, Prentice Hall Pub

II. **Course Description:**

This course utilizes software engineering principles and techniques for the implementation, testing and maintenance of high-quality complex software systems. As designed in a previous course, (CSIT 315)

III. Prerequisite:

CSIT 315 (Application Requirements Definition, Application Requirements Analysis, Software Requirements Specification., Software Design).

IV. **Contents**:

- Software Development process
- Software Testing and Testing Techniques, Tools
- Design – Strategies, tools, patterns
- User Interface Design
- Software Quality and Quality Management
- Risk Analysis and Management
- Cost Estimation
- Project Management
- Software Maintenance
- Ethics
- Evaluating, improving and predicting Product, Process, and Resources
- Reliability

V. Upon completion of the course students should be able to:

- Define what is meant by software engineering and describe the differences between computer science and software engineering.
- Define what is meant by the term “process” and how it applies to software development.
- Describe how process models and project management fit together.
- Describe the activities, resources and products involved in the software development process. Describe several different models of the software development process and understand their drawbacks and when they are applicable..

- Apply several effort and schedule estimation models.
- Identify risks and demonstrate an understanding of what is meant by risk management.
- Explain how and why requirements reviews should be done to ensure quality;
- Explain the difference between a conceptual design and technical design, and the reasons why each is useful for software development;
- Describe an overview of important design styles, techniques and tools, and the conditions under which different choices may be appropriate;
- Identify the characteristics of a good design;
- Explain why validating designs is necessary, and a general overview of how this task can be accomplished;
- Define different types of faults and how to classify them.
- Explain the purpose of testing.
- Explain unit testing and integration testing and explain the differences between them.
- Describe several different testing strategies and understand their differences.
- Describe the purpose of test planning. Develop a test plan, and generate appropriate test cases
- Describe several techniques for determining when to stop testing.
- Describe how system testing differs from unit and integration testing.
- Classify tests as function testing, performance testing, acceptance testing or installation testing.
- Demonstrate an understanding of the purposes and roles of functional testing, performance testing, acceptance testing, and installation testing.
- Define software reliability, maintainability and availability.
- Describe different techniques for measuring reliability, maintainability and availability
- .List the different types of test documentation and know what items belong in test documentation.
- Demonstrate understanding of special problems associated with testing safety-critical systems.
- Describe the principles of Cleanroom and how it differs from conventional testing.
- Describe and produce the types of documentation needed for training.
- Implement and test a software system
- Estimate development time and cost
- Produce high quality artifacts for a software system
- Discuss software engineering ethics

VI. Requirements for completion of the course

i) successful implementation, and testing a previously designed software system.
ii) Satisfactory achievement on tests

Project Presentation: Sec 01 Thur. April 30, 2015 (Note: See the requirements stated
Sec 02 Thur. April 30, 2015 in the Due Dates Document)

Test Dates (Subject to change) Sec 01: Thur March 5th, Thur April 9th
Sec 02: Thur March 5th, Thur April 9th.

Final Exam: Sec 01: Mon May 11, 2015: 3:15 pm-5:15 pm
Sec 02: Thur. May 7, 2015: 5:30 pm-7:30 pm

The grading is based on a point system according to the following:

Points	> 950	900 – 949	850 – 899	800 – 849	750 – 799	700 – 749	650 – 699	600 – 649	550 – 599	500 - 549	< 500
Grade:	A	A-	B+	B	B-	C+	C	C-	D+	D F	

Points Allocation:

Tests	250
Individual Assignments	100
Projects	450
Final Exam	200

v. **Attendance Policy**:

Attendance is mandatory. Since you will be working in teams you need to be considerate of your team members. More than two unexcused absences will affect your final grade. For each unexcused absences beyond two, your grade will be dropped one grade point (A → A-, B → B-, etc). If you have more than five absences you will be dropped from the class. Make-up of missed work is your responsibility. **There are no make-up exam**.

Lateness:

Being late for class on more than four occasions will result in the lowering of your grade

vi. **Academic Dishonesty** (includes, but not limited to the following)

i. The submission of work as one's own which was done by someone else
ii. Cheating on exams or tests: To give or receive assistance in any form during a test or an examination

There is severe penalty for engaging in academic dishonesty

NOTE:

- **The use of cell phones in the class is prohibited. All cell phones must be turned off or be placed in vibration mode**
- Check **Blackboard** at least three times per week

References

- Booch, G. Software Engineering with Ada. Addison-Wesley 1997

- R. Pressman, Software Engineering: A Practitioner's Approach, Addison Wesley, Fifth Edition McGraw-Hill 2001

- A. Davis, Software Requirements: Objects, functions, and States, Prentice-Hall Pub.

- S. Schach, Object-Oriented and Classical Software Engineering, McGraw Hill
- Sommerville, I. Software Engineering, 9th Edition Addison-Wesley, 2011
- Davis, A. M. Software Requirements Analysis and Specification, Prentice-Hall 1990
- Sherman, M. Software Engineering: Reliability, Development and management
- Frederick P. Brooks, Jr., The Mytthical Man-Month *Addison-Wesley* 1975.
- Eric J Braude, *Software Engineering An Object Oriented Perspective John Wiley &sons inc.* 2001
- Eric Gamma,Richard Helm, Ralph Johnson and John Vlissides, *Design Patterns,* Addison-Weseley *1994*

Overview

Software engineering (s.e) is concerned with long-term, large-scale programming projects. This course introduces the topic through lectures and by giving you a chance to help design, manage, and test a medium-sized project. The lectures and the group project will cover topics in s.e. management, problem specification and analysis, system design techniques, documentation, system testing and performance evaluation, reliable software, current design tools and techniques.

Homework/Assignments

Most homework will be related to the course project (described below). There will also be reading assignments and occasional other exercises. You are expected to complete the readings in advance, and relate them to the lecture material. There is an assignment write-up for each step of the project that includes a reading assignment. The readings should be done before you do the assignment. You are encouraged to read ahead as much as possible. Another course requirement is a weekly log of how much time you have spent on different activities related to the course. A simple log sheet is attached. You should turn in your log sheet each week at your last class meeting.

Canvas

Make use of Canvas in communicating with me. For each subsequent class meeting you are to post your questions/concerns regarding the previous class and/or project - difficulties, concept clarification, concept/idea you found particularly interesting, etc. **Be sure to check the Canvas announcements/discussion at least three times per week.**

Team Project

The class will be divided into teams of 4 (!!!) You are strongly encouraged to form your own teams (Regrouping by the instructor is a possibility.) Projects normally will be real-world projects for real clients, and thus will be utilized at its completion.. Each team will have regular weekly meetings (with a consultant). The project(s) will take most of the semester with major write-ups due at approximately two week intervals. There will also be some oral presentations, including a final demonstration of the project design.

Tests and Individual Projects

There will be two in-class exams and a final exam

Grades

Forty five percent (45%) of your grade will be based on the team project, documents and demonstration. You are graded on the quality of the work you produce, not on the number of hours a week you spend. Use your energy, and time, wisely. The rest of the grade will depend on individual tests, assignments, and contributions to the group project. It's very important that project assignments be turned in promptly, both to allow you time to 'complete' the tasks on schedule and allow time for evaluation of your work and for "suggestions!!!" to be made.

Keeping a Log

Each person is to maintain a log of the time spent on course work and to break down the total time into categories. For example, categories might include time spent reading textbooks or articles, reading project problem descriptions, devising specifications, testing your program, etc. Hopefully, these logs will help you see how you spend your time and help you to make better predictions of the time needed by the different phases of a software project.

When in doubt, use as many categories as occur to you

Name: ________________________________ Date: _______________

Instructor: ______________________________ Assignment: _________________

Phase: _______________________________________

Date	Start Time	Activity	End	Elapsed Time	Comments

Activity Log