Software Development Practices  
COSC 6397 (Dr. Jaspal Subhlok)  
COSC 4397 (Dr. Venkat Subramaniam)

Course Description: The objective of this course is to bring the students up to speed on the pragmatics of software development. In this hands-on course, the students will work towards developing industrial strength software systems using state of the art techniques and tools. Emphasis will be placed on estimation, planning, risk management, testing, development and deployment. Various good practices in software development, including pragmatic programming and agile software development practices will be emphasized.

Pre-requisite:
- You must have completed either COSC 6318 or COSC 4351
- You must have completed COSC 1320 or equivalent
- You must have completed COSC 2320 or equivalent

Pre-requisites will be enforced (expected to have knowledge of OO, Data Structures, and software development life cycle). If you have any questions about them, please contact Venkat. Please fill the pre-requisite certification form available on the course web site and submit it to Venkat by 6PM August 23rd.

Topics:

Agile Software Development  
Principles and practices  
Tools for agile development  
Unit Testing  
Continuous integration  
Continuous Feedback  
Estimation  
Measuring progress  
Measuring Quality

Course organization: There will be a few lecture (3 or 4). After that the course will be driven by student project iteration. There will be a project progress review every two weeks with deliverables at the middle and end of the semester.

Lecture Schedule:
August 21 - Introduction to course, expectations, process, project expectations, and student evaluations  
August 23 – Project team formation  
August * - Online tape viewing – Unit Testing, Mock Objects, Refactoring  
August 30 - Continuous Integration  
September 13 – Agile Methodologies

Project: Each student will participate in a software project. The instructors will play the role of customers in terms of defining the (evolving) requirements, evaluating the progress of the work, and deciding if it meets the requirements. The success of the project is measured based on a number of criteria including (but not limited to) working software delivered, continued progress and pace through out the semester, process and practices followed, quality of the code, ability to accommodate reasonable change in requirements, feature completeness, progress in terms of
schedule, use of tools, techniques, practices and facilities that reduce risk and promote the probability of success.

Project Team Requirements:
• Each team can have at most 6 students
• Each student is required to be part of one and only one team
• Each project team will have a team lead

*Any exceptions from these requirements require the approval of the instructor.*

Project: Each group of students have to define their own project and have them approved by the instructors. Real life projects from research groups or industry are encouraged but not required (Such projects still must be approved by instructors and follow all the class guidelines).

Project Schedule:
• August 23rd Team formation – submit names of students in your team (plus an optional nickname for your team)
• September 6th – Final day to email a two page document for the proposal (students are expected to consult with the instructors well before this day about their ideas)
• September 11th – Each team should make a ten minute presentation on what their project is
• September 18th – Teams start their two week iteration demo.
• October 16th – Submit first incremental deliverable
• November 26th – November 29th final demo of the project and deliverable

Grading:
• Grades for project components will depend on the overall group effort as well as the individual contribution of students.
  o Iteration demo and progress 50%
  o First incremental deliverable 10%
  o Final deliverable 10%
  o Final demo 10%
  o Student evaluation of project 5%
  o Individual Peer review 5%
  o Individual student report (details provided later) and interview 10%

*Some details of grading may change but this should give you a good idea.*

Recommended Text: There are no required texts for this course. Some recommended texts are:
• Agile Software Development: Principles, Practices, and Patterns by Robert Martin
• Pragmatic Project Automation: How to Build, Deploy, and Monitor Java Apps by Mike Clark
• Test Driven Development: By Example by Kent Beck
• Pragmatic Unit Testing by Dave Thomas and Andy Hunt