EXPANDED COURSE DESCRIPTION
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
Lassonde School of Engineering
Electrical Engineering Computer Science
LE / EECS 4314 3.0 SECTION E
ADVANCED SOFTWARE ENGINEERING
FALL 2017 / WINTER 2018

Last Modified Date: 08/18/2017

COURSE CALENDAR DESCRIPTION

Advanced software engineering methods, processes and measurements (from requirements through to system evolution) for managing and delivering reliable software on time and within budget. Understanding the different types of systems and their associated technical, ethical and professional issues. Three lecture hours per week. One term. Three credits. Prerequisites: General prerequisite; LE/EECS 2030 3.00 or LE/EECS 1030 3.00; LE/EECS 3311 3.00. (NOTE: The General Prerequisite is a cumulative GPA of 4.50 or better over all major EECS courses. EECS courses with the second digit “5” are not major courses.)

INSTRUCTOR(S)

<table>
<thead>
<tr>
<th>Name</th>
<th>Section / Format / Term</th>
<th>Contact Email</th>
<th>Contact Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jiang, Zhen Ming</td>
<td>Sec. E / LECT / F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ADDITIONAL INFORMATION

Calendar Descriptions for this Course:

This course goes into more detail about some of the software engineering techniques and principles presented in earlier courses, as well as introduces advanced aspects of software engineering that are not addressed elsewhere:

- Software process and its various models and standards (CMMI, ISO 9001).
- Software architecture, i.e. the structure of data and program components that are required to build a software system. Examples include distributed and component-based architectures.
- Model Driven Engineering and the use of software description languages.
- Software metrics, such as metrics for software quality, software design metrics, as well as testing and maintenance metrics.
- Project management concepts on coordinating people and products.
- Cost estimation and project scheduling for large software systems.
- Risk management and mitigation.
- Software configuration management (software evolution, change management, version and release management).
- Emerging technologies, such as security engineering, service-oriented software engineering, and aspect-oriented software development.

LEARNING OUTCOMES

- Derive models of software systems and express them in a language such as UML.
- Understand the differences between different types of software architecture
- Apply metrics that estimate the quality, maintainability, and test adequacy of a software system.
- Derive cost estimation tables delineating the tasks to be performed, and the cost, effort, and time involved for each task.
- Identify risks associated with a given software project, and develop plans to mitigate and manage these risks.
• Manage software projects by identifying the sequence of tasks that will enable the project to complete in time, assigning responsibility for each task, and adapting the schedule as various risks become reality

GRADED ASSESSMENT
The weight distribution of the course components is as follows:
60% - Assignments
40% - Final Exam

ACADEMIC INTEGRITY LINKS
• Senate Policy on Academic Honesty - http://secretariat-policies.info.yorku.ca/policies/academic-honesty-senate-policy-on/
• Academic Integrity - http://lassonde.yorku.ca/academic-integrity

STUDENT LINKS
• Student Rights and Responsibilities - http://oscr.students.uit.yorku.ca/student-conduct
• Religious Observance - https://w2prod.sis.yorku.ca/Apps/WebObjects/cdm.woa/wa/regobs
• Academic Accommodation for Students with Disabilities - http://secretariat-policies.info.yorku.ca/policies/academic-accommodation-for-students-with-disabilities-policy/
• Counselling and Disability Services - http://cds.info.yorku.ca/

Many courses utilize Moodle, York University's course website system. If your course is using Moodle, click here to access it.
Moodle @ York University