Computing Science Course Outlines  2018 Summer

CMPT 475 - E100 Requirements Engineering

Instructor(s): Janice Regan

Calendar Objective/Description:
Software succeeds when it is well-matched to its intended purpose. Requirements engineering is the process of discovering that purpose by making requirements explicit and documenting them in a form amenable to analysis, reasoning, and validation, establishing the key attributes of a system prior to its construction. Students will learn methodical approaches to requirements analysis and design specification in early systems development phases, along with best practices and common principles to cope with notoriously changing requirements.

Instructor's Objectives:
Software requirements involve both design and understanding of what is needed by the application. This is a creative activity that calls for abstract models to analytically analyze and to reason out requirements. Design decisions and conformance criteria, making sure these are well understood prior to coding. Starting with software requirement analysis methodologies, abstraction principles and specification paradigms. Students will learn how to use modeling as an effective instrument for making software systems more reliable, the requirements gathering process more predictable, and overall improve the quality of the resulting product. Students will evaluate contrasting methodologies and how to ensure high quality requirements be provided to the development stage of software engineering process.

Prerequisites:
CMPT 275 or 276, MACM 201 and 15 units of upper division courses. Recommended: co-op experience.

Topics:
- Requirements position in the Software Development Lifecycle (SDLC)
- Scoping
- Methods for requirements analysis
- Design Models
- Functional and Non-Functional Requirements
- Risk Identification and Management
- Requirements Management
- Completeness and Consistency checking
- Formal Specifications
- Importance of User Community to Requirements Engineering

Grading:
assignments/project 50% midterm 20% final 30%
Students must attain an overall passing grade on the weighted average of exams in the course in order to obtain a clear pass (C- or better).

Reference Books:
Software Engineering, 10th Edition, Ian Sommerville, Addison-Wesley, 2015, 9780133943030, General reference to basic software engineering principles

Facts and Fallacies of Software Engineering, Robert L. Glass, Addison-Wesley, 2003, 9780321117427

The Logic of Failure, Dietrich Dorne, Basic Book, 1996, 9780201479485

Code Complete, Steve McConnell, Microsoft Press, 2004, 9780735619678

**Academic Honesty Statement:**

Academic honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty will be handled in accordance with the SFU Academic Honesty and Student Conduct Policies (http://www.sfu.ca/policies/gazette/student.html).