SWEN90006 Software Engineering Methods

Credit Points:	12.50
Level:	9 (Graduate/Postgraduate)
Dates & Locations:	2010, Parkville This subject commences in the following study period/s: Semester 2, Parkville - Taught on campus.
Time Commitment:	Contact Hours: 24 one-hour lectures (two per week) and 12 one hours workshops (one per week) Total Time Commitment: 120 hours
Prerequisites:	433-320 Software Modelling and Design
Corequisites:	None
Recommended Background Knowledge:	None
Non Allowed Subjects:	433 342 Software Engineering
Core Participation Requirements:	For the purposes of considering request for Reasonable Adjustments under the Disability Standards for Education (Cwth 2005), and Students Experiencing Academic Disadvantage Policy, academic requirements for this subject are articulated in the Subject Description, Subject Objectives, Generic Skills and Assessment Requirements of this entry.The University is dedicated to provide support to those with special requirements. Further details on the Disability support scheme can be found at the Disability Liaison Unit Website:http://www.services.unimelb.edu.au/disability/
Coordinator:	Dr Timothy Miller
Contact:	Melbourne School of Engineering Office Building 173, Grattan Street The University of Melbourne VIC 3010 Australia General telephone enquiries + 61 3 8344 6703 + 61 3 8344 6507 Facsimiles + 61 3 9349 2182 + 61 3 8344 7707 Email **eng-info@unimelb.edu.au (mailto:eng-info@unimelb.edu.au)**
Subject Overview:	This subject introduces students to the software engineering principles, processes, tools and techniques for analysing, measuring and developing quality and dependable software. The subject is one of the two SE foundational subjects and looks at methods and tools for analysing, measuring, and building dependable software systems. Topics covered include: methods for static and dynamic software testing; quality and dependability; reliability engineering; performance engineering; fault tolerance; software problem analysis and fault isolation; and software tools.
Objectives:	On completion of this subject, students should be able to: # Select appropriate methods to build in quality and dependability into software systems # Select and apply a effective testing techniques for verifying medium and large scale software systems; and # Select and apply measures and models to evaluate the quality and dependability of a software system.
Assessment:	Project work during semester, expected to take about 36 hours (50%); and a 2-hour end-of-semester written examination (50%). To pass the subject, students must obtain at least 50% overall, 25/50 in project work, and 25/50 in the written examination.

Prescribed Texts:	TBA
Breadth Options:	This subject potentially can be taken as a breadth subject component for the following courses: # **Bachelor of Arts (https://handbook.unimelb.edu.au/view/2010/B-ARTS)** # **Bachelor of Commerce (https://handbook.unimelb.edu.au/view/2010/B-COM)** # **Bachelor of Environments (https://handbook.unimelb.edu.au/view/2010/B-ENVS)** # **Bachelor of Music (https://handbook.unimelb.edu.au/view/2010/B-MUS)** You should visit **learn more about breadth subjects (http://breadth.unimelb.edu.au/breadth/info/index.html)** and read the breadth requirements for your degree, and should discuss your choice with your student adviser, before deciding on your subjects.
Fees Information:	Subject EFTSL, Level, Discipline & Census Date, http://enrolment.unimelb.edu.au/fees
Generic Skills:	On completion of this subject students should have developed the following generic skills: # An in-depth technical competence in the selection and application of methods to develop, measure and test quality of software systems # The ability undertake problem identification, formulation and solution.
Related Course(s):	Bachelor of Engineering Bachelor of Engineering (Computer Engineering) Bachelor of Engineering (Electrical Engineering) Bachelor of Engineering (IT) Computer Engineering Bachelor of Engineering (IT) Electrical Engineering Bachelor of Science
Related Majors/Minors/ Specialisations:	Computer Science Computer Science Software Systems