LINGI2252
2016-2017

Software Maintenance and Evolution

5.0 credits	30.0 h + 15.0 h	1q

Teacher(s) :	Mens Kim ;
Language :	Anglais
Place of the course	Louvain-la-Neuve
Inline resources:	> http://icampus.uclouvain.be/claroline/course/index.php?cid=INGI2252
Main themes :	Whereas many software engineering courses focus on building new systems from scratch, in industrial practice software developers are often confronted with already existing software systems that need to be maintained, reused or evolved. This requires specific skills to understand the design and implementation of an existing system and which parts need to be modified, to build software systems that are easier to maintain, and to design systems with reuse and evolution in mind from the very start. This course will thus study a variety of techniques, tools and methodologies to help building software systems that are easier to understand, maintain, reuse and evolve: -- Software development in the context of an existing code base as opposed to 'green field' development o Software comprehension and concern location o Change impact analysis o Reverse engineering -- Software Maintenance o Best programming practices o Coding standards o Design principles and heuristics o Design patterns o Refactoring o Reengineering -- Software Reuse and Evolution o The laws of software evolution o Reuse techniques and design for reuse o Libraries vs. application frameworks -- Software product lines
Aims :	Given the learning outcomes of the "Master in Computer Science and Engineering" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: -- INFO1.1 , INFO1.3 -- INFO2.5 -- INFO5.5 Given the learning outcomes of the "Master [120] in Computer Science" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: -- SINF1.M3 -- SINF2.5 -- SINF5.5 Students completing successfully this course will be able to -- Understand the difficulties of developing code in a change context as opposed to 'green field' development -- Assess the impact of a change request to an existing product of medium size. -- Describe techniques, coding idioms and other mechanisms for implementing designs that are more maintainable. -- Understand how design patterns can improve the design of a software system. -- Refactor an existing software implementation to improve some aspect of its design. --

	Identify the principal issues associated with software evolution and explain their impact on the software lifecycle.' -- Discuss the advantages and disadvantages of different types of software reuse. *The contribution of this Teaching Unit to the development and command of the skills and learning outcomes of the programme(s) can be accessed at the end of this sheet, in the section entitled "Programmes/courses offering this Teaching Unit".*
Content :	The course will be organized in 3 different modules that address a variety of techniques, tools and methodologies to help building software systems that are easier to understand, maintain, reuse and evolve: 1. Software understanding : how to understand existing systems 2. Software maintenance : how to build software that is more maintainable, or how to improve existing software to make it more maintainable 3. Software reuse and evolution : how to build reusable software and how to reuse existing software
Bibliography :	References Given the variety of topics covered this course will not follow a single textbook but will be based on material from many different sources. As such, the course slides will be the main reference material for this course and pointers to additional reading material will be provided through the online course platform. Support The course slides as well as other relevant and practical information related to the course will be accessible on Moodle. The same platform will also be the means of communication between the teacher(s) and the students.
Other infos :	Even though good quality software may be easier to maintain and evolve, software quality assurance techniques will not be addressed explicitly in this course as they are the topic of a separate course on Software Quality Assurance [LINGI2251] Background: -- Having a good knowledge of and experience with object-oriented programming concepts, algorithms and data structures. -- Having prior or simultaneous experience with the development of a medium- to large-scale software system.
Faculty or entity in charge:	INFO

Programmes / formations proposant cette unité d'enseignement (UE)

Intitulé du programme	Sigle	Credits	Prerequis	Acquis d'apprentissage
Master [120] in Computer Science	SINF2M	5	-	🔍
Master [120] in Computer Science and Engineering	INFO2M	5	-	🔍