UCL Université catholique de Louvain

LINGI2252

2014-2015

Software Engineering : Measures and Maintenance

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 :	-- "Best practices" of object-oriented programming ; -- Reuse techniques and application frameworks ; -- Software measures and metrics ; -- Software comprehension and reverse engineering ; -- Software reengineering and refactoring and restructuring ; -- The use of variety of tools that support some of the above activities.
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: -- INFO1.1-3 -- INFO2.3, INFO2.5 -- INFO5.3-5 -- INFO6.1, INFO6.3, INFO6.4 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 -- SINF2.3, SINF2.5 -- SINF5.3-5 -- SINF6.1, SINF6.3, SINF6.4 Students completing successfully this course will be able to -- analyze the quality of a software system (and more specifically, its maintainability) ; -- understand the nature of some of the problems encountered when maintaining complex software systems ; -- suggest appropriate solutions to improve reusability and maintainability of a software system, measure its quality and support its evolution ; -- program in Smalltalk, a pure object-oriented programming language. Students will have developed skills and operational methodology. In particular, they have developed their ability to -- have a critical look at the results of an automated tool; -- make a persuasive presentation; -- write a summary report. *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".*

Evaluation methods :	The course evaluation is based on the reports and presentation of their work on the case study. Each student will carry out a small project during the semester, and summarise the results of this project in a scientific report and/or presentation. The final mark of this course will be based on the results reported on, the quality of the report and (if necessary) an oral project defence organised during the exam session. Conditions for participating in the exam (= project defense) in June or September: having submitted the project report within the deadline imposed by the teacher. For the exam of September, the mark obtained in June will not be considered. However, the student will be asked to conduct a new project (and write an associated report) of the same complexity and size as in June.
Teaching methods :	The theoretical aspects are introduced in the theory sessions. A case study application will be analyzed using the techniques seen in the course by groups of 2. The students are asked to analyze the qualities of this application (and its maintainability in particular) and suggest possible improvements to that application. The results of their work will be summarised in intermediate and final reports and during a presentation. The students are asked to write the reports and present their work in English.
Content :	The theoretical aspects that will be introduced in the theory sessions will be put in practice during "hands-on" practical sessions in one of the computer rooms. A single software application will be developed throughout the different practical sessions, and the techniques taught in the theory course will be tested on this application. The course evaluation will be an assignment where the students are asked to apply the learned techniques on a software application of their choice, more specifically to analyze the qualities of this application (and its maintainability in particular) and suggest possible improvements to that application.
Bibliography :	The theory course relies on several books, such as : -- N.E. Fenton and S.L. Pfleeger, " Software Metrics: A Rigorous and Practical Approach", 2nd edition, Thomson Computer Press, 1996. -- K.Beck, "Smalltalk Best Practice Patterns", Addison-Wesley, Prentice Hall, 1996 -- M. Fowler, "Refactoring, Improving the Design of Existing Code", Addison-Wesley, 1999 -- A. Black, S. Ducasse, O. Nierstrasz, "Pharo By Example", 2009. -- A. Riel, "Object-Oriented Design Heuristics", Addison-Wesley, 1996. The course slides as well as the practical session guides and other practical information related to the course will be accessible online.
Other infos :	Background: -- LSINF1121 : Object-oriented programming and experience with medium-size codes -- LSINF2255/LINGI2255 : simultaneous or previous experience in developing large-size software systems.
Cycle and year of study :	> Master [120] in Computer Science > Master [120] in Computer Science and Engineering
Faculty or entity in charge:	INFO