

5.0 crédits

30.0 h + 15.0 h

1q

Enseignants:	Mens Kim ;
Langue d'enseignement:	Anglais
Lieu du cours	Louvain-la-Neuve
Ressources en ligne:	> http://www.icampus.ucl.ac.be/LINGI2252/
Préalables :	-- programmation orienté-objet et expérience avec des codes de taille moyenne (p.e. SINF1121) -- avoir en parallèle ou précédemment un expérience de développement d'un logiciel de grande taille (p.e. INGI2251 et SINF2255/INGI2255)
Thèmes abordés :	-- Les "bonnes pratiques" de la programmation orienté-objet; -- Les techniques de réutilisation, application frameworks; -- Les métriques et mesures de logiciels; -- La gestion de versions: variantes, révisions, configurations; -- La compréhension et rétro-ingénierie de logiciels; -- La ré-ingénierie et la restructuration de logiciels; -- L'utilisation de certains outils qui supportent les taches mentionnées ci-dessus.
Acquis d'apprentissage	Les étudiants ayant suivi avec fruit ce cours seront capables de -- expliquer comment mesurer les qualités d'un logiciel (et notamment la qualité de la maintenabilité); -- utiliser divers outils pour mesurer les qualités d'un logiciel et analyser de façon critique les résultats obtenus en les comparant avec ceux provenant d'autres outils et en inspectant le logiciel manuellement pour confirmer ou contredire les résultats obtenus; -- expliquer la nature des problèmes rencontrés dans la maintenance de logiciels complexes et les processus impliqués; -- mettre en oeuvre quelques approches-type pour favoriser la réutilisabilité et la maintenabilité; -- programmer des applications maintenables et réutilisables en Smalltalk, un langage orienté-objet pur. Les étudiants auront développé des compétences méthodologiques et opérationnelles. En particulier, ils ont développé leur capacité à -- avoir un regard critique sur les résultats d'un outil automatisé; -- réaliser une présentation convaincante; -- rédiger un rapport de synthèse. <i>La contribution de cette UE au développement et à la maîtrise des compétences et acquis du (des) programme(s) est accessible à la fin de cette fiche, dans la partie « Programmes/formations proposant cette unité d'enseignement (UE) ».</i>
Modes d'évaluation des acquis des étudiants :	L'évaluation du cours est basé sur les rapports et la présentation de leurs travaux sur l'étude de cas. Chaque étudiant devra réaliser un petit projet au cours du quadrimestre, avec un rapport à produire. La note finale se basera sur les résultats du projet, la qualité du rapport, et (si nécessaire) une entrevue organisé pendant la session. Conditions pour pouvoir participer à l'examen (= défense de projet) de juin / septembre : avoir soumis le rapport dans l'échéance indiqué par le professeur. Pour l'examen de septembre la note de juin n'interviendra plus. Or, l'étudiant devra finaliser un nouveau projet du même ampleur que pour la session de juin.
Méthodes d'enseignement :	Les aspects théoriques sont introduits dans les séances de théorie. Ils sont mis en pratique pendant les travaux pratiques sur machine dans l'une des salles informatiques. Un seul logiciel sera développé tout au long des différentes sessions pratiques et les techniques enseignées dans le cours théorique seront testées sur ce logiciel. Une étude de cas sera réalisée sur un logiciel à analyser en utilisant ces techniques, par groupe de 2. Les étudiants sont invités à analyser les qualités de cette application (et notamment sa maintenabilité) et à suggérer des améliorations possibles. Les résultats de leur travail sera expliqué dans des rapports intermédiaires et final et lors d'une présentation. Les étudiants sont invités à écrire les rapports et à présenter leurs travaux en anglais.
Contenu :	La théorie introduite dans les exposés magistraux sera mise en pratique pendant les travaux pratiques qui se dérouleront dans une salle informatique. Un seul logiciel sera développé au cours des différentes séances des travaux pratiques, et les différentes techniques apprises seront testées sur ce logiciel. L'évaluation de ce cours est un projet ou les étudiants doivent appliquer les techniques apprises sur un logiciel au choix, afin d'analyser les qualités de ce logiciel (et notamment sa maintenabilité).

<p>Bibliographie :</p>	<p>La théorie du cours se base sur plusieurs livres :</p> <ul style="list-style-type: none"> -- 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 <p>Les transparents des cours magistraux ainsi que les énoncés des séances pratiques et d'autres informations pratiques relatives au cours seront accessibles en ligne.</p>
<p>Autres infos :</p>	
<p>Cycle et année d'étude :</p>	<p>> Master [120] : ingénieur civil en informatique > Master [120] en sciences informatiques</p>
<p>Faculté ou entité en charge:</p>	<p>INFO</p>