UCL Université catholique de Louvain

# LINGI2251 Software engineering: development methods

*2013-2014*

| | | |
|---|---|---|
| 5.0 credits | 30.0 h + 30.0 h | 2q |

| | |
|---|---|
| Teacher(s) : | Pecheur Charles ; |
| Language : | Anglais |
| Place of the course | Louvain-la-Neuve |
| Inline resources: | > http://icampus.uclouvain.be/claroline/course/index.php?cid=LINGI2251 |
| Prerequisites : | -- <br> mathematical logic as provided by the courses INGI1101 <br> -- <br> master of object-oriented programming, algorithms and data structures as provided by the SINF1121 <br> -- <br> participating in the implementation of a small-size software project (for example SINF1124) |
| Main themes : | -- <br> The what, why and how of software engineering; <br> -- <br> Software life-cycle phases (introduction to requirements engineering, software architecture and design, software construction, software verification and validation, deployment, software evolution and maintenance); <br> -- <br> Software process models (e.g., waterfall, incremental, agile); <br> -- <br> Software process capability maturity models; <br> -- <br> Software quality concepts and quality assurance; <br> -- <br> Software modelling (requirement models, functional, architectural and behavioural models); <br> -- <br> Program specification and correctness. <br> -- <br> Software verification and validation, testing. |
| Aims : | Students completing successfully this course will be able to: <br> -- <br> Describe main issues raised by large software projects; <br> -- <br> Master systematic approaches to software development; <br> -- <br> Explain the concept of a software life-cycle and provide an example, illustrating its phases, including the various products that are produced; <br> -- <br> Assess the impact of the decisions made at various phases of this lifecycle; <br> -- <br> Compare common software process models with respect to their value for developing particular classes of software systems; <br> -- <br> Explain the purpose of process maturity models and determine the maturity of a given process; <br> -- <br> Describe fundamental challenges of and common techniques used for requirements engineering and apply them for a simple software system; <br> -- <br> Define software quality, discuss the various qualities that a large-scale complex software application should exhibit and describe the role of quality assurance activities in the software process; <br> -- <br> Model rigorously the design of a product in order to ensure the quality of the final product and its development process; <br> -- <br> For a simple software system, propose an appropriate software architecture and discuss its advantages and disadvantages; <br> -- <br> Specify program components and use these specifications to verify their correct behaviour. <br> -- <br> Apply a variety of testing and other verification techniques to verify program correctness. |

| | |
|---|---|
| | *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 evaluation will consist of the assignments and quizzes (50%) and an oral exam covering the lectures (50%). The assignments and quizzes can be presented again in the second session. |
| Teaching methods : | The course will consist of lectures and lab sessions.<br><br>One or two lectures will consist of presentations by industry specialists.<br><br>In the lab sessions, students will apply software engineering techniques and tools to sample software applications.<br><br>During the quadrimester, students will have to complete several assignments and/or quizzes, applying the tools and techniques seen in the lab sessions. |
| Content : | Software engineering is the discipline concerned with the application of theory, knowledge, and practice to effectively and efficiently build reliable software systems that satisfy the requirements of customers and users. This discipline is applicable to small, medium, and large-scale systems. It encompasses all phases of the lifecycle of a software system, including requirements elicitation, analysis and specification; design; construction; verification and validation; deployment; and operation and maintenance. Whether small or large, following a traditional disciplined development process, an agile approach, or some other method, software engineering is concerned with the best way to build good software systems. |
| Bibliography : | Shari Lawrence Pfleeger and Joanne Atlee, Software Engineering: Theory and Practice, 4th edition, Pearson, 2010.<br>Lectures slides will be available on the course website. |
| Cycle and year of study : | > Master [120] in Computer Science and Engineering<br>> Master [120] in Computer Science<br>> Master [120] in Biomedical Engineering |
| Faculty or entity in charge: | INFO |