CS-305

Software engineering

Candea George

<table>
<thead>
<tr>
<th>Cursus</th>
<th>Sem.</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>HES - IN</td>
<td>H</td>
<td>Obl.</td>
</tr>
<tr>
<td>Informatique</td>
<td>BA5</td>
<td>Obl.</td>
</tr>
<tr>
<td>Mineur en Informatique</td>
<td>H</td>
<td>Opt.</td>
</tr>
<tr>
<td>Science et ing. computationelles</td>
<td>MA1, MA3</td>
<td>Opt.</td>
</tr>
<tr>
<td>Systèmes de communication</td>
<td>BA5</td>
<td>Opt.</td>
</tr>
</tbody>
</table>

Summary
Covers basic aspects of modern software development practices and tools, foundations of software engineering, how to think about software, structure it, modify it, and improve it. Covers the software development process (including agile methods) and working as part of a team of developers.

Content

- Object-oriented design and reasoning
- Design patterns
- Principles of building reliable and secure software
- Performance tuning
- Testing and debugging
- Code layout and style
- Development processes
- Software project management
- Tools for writing, analyzing, and debugging code, as well as source code management

Being a good software engineer entails a continuous learning process. Unlike mathematics or physics, this field changes fast, thus making continuous and independent learning essential. This course prepares students to become lifelong auto-didacts that build upon the foundation of immutable principles governing good software engineering.

Keywords
software development, software engineering, software design, software development tools, development processes, agile methods

Learning Prerequisites

Required courses
First and foremost, students taking Software Engineering must be proficient Java programmers. Without a good prior knowledge of Java, it is very difficult to keep up with the pace of the class. Furthermore, this course builds on material taught in the courses below. Students who do not master the material taught in the prerequisite courses prior to starting Software Engineering typically do not manage to pass:

- CS-108 Practice of Object-Oriented Programming
- CS-210 Functional Programming
- CS-206 Parallelism and concurrency
- CS-207 System-oriented Programming
Recommended courses
The material in the following courses is helpful but not absolutely required:
• COM-208 Computer networks
• CS-208/209 Computer architecture

Important concepts to start the course
• Good knowledge of object-oriented programming (e.g., in Java)
• Knowledge of using version control systems (e.g., Git)
• Using modern development tools (e.g., IDE, Android emulator)

Learning Outcomes
By the end of the course, the student must be able to:
• Design software that is reliable, secure, user-friendly, and performs well
• Implement (in software) sophisticated designs and algorithms
• Specify requirements for software systems
• Develop code that is maintainable
• Organize a team of developers to execute a medium-sized software project
• Assess / Evaluate design and implementation options
• Choose alternatives to optimize for an objective (e.g., performance)

Transversal skills
• Plan and carry out activities in a way which makes optimal use of available time and other resources.
• Set objectives and design an action plan to reach those objectives.
• Assess progress against the plan, and adapt the plan as appropriate.
• Manage priorities.
• Evaluate one's own performance in the team, receive and respond appropriately to feedback.
• Give feedback (critique) in an appropriate fashion.
• Resolve conflicts in ways that are productive for the task and the people concerned.
• Assess one's own level of skill acquisition, and plan their on-going learning goals.
• Identify the different roles that are involved in well-functioning teams and assume different roles, including leadership roles.

Teaching methods
• Combination of online and in-class lectures
• Recitations and lab sessions
• Homework exercises
• Course project

Expected student activities
• Watch online lectures
• Attend and participate in lectures and recitations
• Read and understand assigned materials
• Complete programming assignments and attend lab sessions
• Work with team members to complete a substantial project

Assessment methods
Throughout the semester (contrôle continu). The final grade will be determined approximately as follows:
• 10% for 2 homework assignments done individually
• 50% for 1 team project
• 40% based on 2 exams during the semester and potentially online quizzes

Exact formula may vary from year to year.

Supervision
Office hours    Yes
Assistants    Yes
Forum         Yes
Others        See http://sweng.epfl.ch/

Resources
Virtual desktop infrastructure (VDI)
No

Bibliography
See http://sweng.epfl.ch for up-to-date bibliography

Ressources en bibliothèque
• Code Complete: A Practical Handbook of Software Construction / McConnell
• The Deadline: A Novel About Project Management / DeMarco

Websites
• http://sweng.epfl.ch/