ADVANCED TOPICS IN SOFTWARE ENGINEERING

- **Course code**

  8138

- **Type of course (compulsory, optional)**

  Compulsory or optional depending on the specialization

- **Level of course**

  First Cycle

- **Year of study**

  2013-2014

- **Semester**

  6th Semester

- **Number of credits allocated**

  6 ECTS Credits

- **Name of lecturer**

  Spinellis Diomidis, Professor

- **Learning outcomes**

  While most Information Systems and Computer Science courses traditionally deal with the development of new systems, in practice developers spend the largest part of their time in software life-cycle activities that follow the development phase. The objective of the course is to allow students to read, understand, and evaluate a system's software elements (code, structure, architecture). Having followed this course, students should be able to intelligently decide on how existing systems will be maintained, setup design and evolution strategies for legacy code, and prescribe the use of refactoring for dealing with architectural mismatches and low-quality code. An innovative aspect of the course involves the use of Open Source Software (OSS) in course examples and exercises. Through the study of OSS students will be able to see how non-trivial applications like the Apache Web server, the Postgres Relational Database Management System, the Jakarta Java servlet container and the Cocoon framework are structured.

- **Mode of delivery (face-to-face, distance learning)**

  Lectures, lab tutorials, and graded coursework

- **Prerequisites and co-requisites**

  Programming I, Programming II

- **Recommended optional programme components**

  Analysis and Design of Information Systems

- **Course contents**

  Code as Part of the Software Development Process; The Open Source Landscape; Tackling Large Projects; General Purpose Tools; Version Control; Build Management; Collaboration; Performance Measurement and Management; Code-Reading Tools; Inspection and Testing; Coding Standards and Conventions; Documentation; Maintainability; Basic Programming Elements

- **Recommended or required reading**

  📖 Diomidis Spinellis. Code Reading: The Open Source Perspective. Addison-Wesley, 2003.

  📖 Diomidis Spinellis. Code Quality: The Open Source Perspective. Addison-Wesley, 2006.

  📖 Martin Fowler. Refactoring: Improving the Design of Existing Code. Addison-Wesley, 2000.

  📖 Michael Feathers. Working Effectively with Legacy Code. Prentice-Hall, Englewood Cliffs, NJ, 2005.

- **Planned learning activities and teaching methods**

  Lectures, coursework, assessed presentations, team-working, contribution to open source projects, collaboration with the projects' developers

- **Assessment methods assessment methods and criteria**

  The course is assessed through the students' contribution to an existing open source project, through the following criteria:

  Understanding and documentation of legacy system (10%)

  Breadth of changes (20%)

  Design quality (10%)

  Implementation quality (10%)

  Integration (10%)

  Testing (10%)

  Coordination with the development team (10%)

  Presentation (10%)

  Documentation and quality of the deliverables (10%)

  Team's blog (10%)

- **Language of instruction**

  Greek / English (when taught to Erasmus students)

- **Work placement(s)**

  N/A