Course Unit Descriptor

Study Programme: Ph.D. in Computer Science

Course Unit Title: Software Evolution

Course Unit Code: ID101

Name of Lecturer(s): Zoran Budimac, Miloš Radovanović

Type and Level of Studies: Doctoral Academic Degree

Course Status (compulsory/elective): Elective

Semester (winter/summer): Winter

Language of instruction: Serbian (primary), English (secondary)

Mode of course unit delivery (face-to-face/distance learning): Face-to-face

Number of ECTS Allocated: 7

Prerequisites: none

Course Aims:

With the emergence of new architectures, need for representing new functionality, improvements in project development techniques and/or changes in goals and business processes, there exists a stron urge for existing software systems to evolve, preserving the continuity of use. Such evolution demands different techniques for what is known as `*re-engineering*'. With re-engineering, we assume the viewpoint of exploring, understanding and changing the system with the aim of redesigning and implementing it in a new form.
The course goal is acquaintance with all aspects of the aforementioned process, and recognising the functionalities of existing code.

Learning Outcomes:

- Critically assess existing basis for software evolution
- Critically assess re-engineering techniques for software migration and abstraction
- Critically assess approaches to software evolution life cycles
- Apply research methods in software evolution

Syllabus:

Theory
Overview of the state of research in the field: evolution within software life cycles, laws of evolution, software transformation, transformation theory and its implementation, software abstraction. Contemporary areas of research in the field, e.g., software quality preservation, unified software platform for evolution, model evolution, formal basis of software evolution, support for multi-language systems, evolution as language construct, etc.
Practice
-

Required Reading:

1. H. Yang, M. Ward. *Successful Evolution of Software Systems*. Artech House, 2003
2. M. Fowler. *Refactoring: Improving the Design of Existing Programs*. Addison-Wesley, 1999
3. S. Demeyer, S. Ducasse, O. Nierstrasz. *Object-Oriented Reengineering Patterns*. Morgan-Kaufmann, 2002

Weekly Contact Hours: 2	**Lectures:** 2	**Practical work: 0**

Teaching Methods:
Lectures are organized using classic teaching methods with use of a projector. Students independently explore various research topics, present and discuss results with other students and the lecturer.

Knowledge Assessment (maximum of 100 points):

Pre-exam obligations	Points 50	**Final exam**	Points 50
Active class participation		written exam	
Practical work		oral exam	

Preliminary exam(s)		…….	
Seminar(s)			
The methods of knowledge assessment may differ; the table presents only some of the options: written exam, oral exam, project presentation, seminars, etc.			