Course Unit Descriptor

Study Programme: Ph.D. in Computer Science

Course Unit Title: Software Testing and Validation

Course Unit Code: ID102

Name of Lecturer(s): Zoran Budimac, Gordana Rakić

Type and Level of Studies: Doctoral Academic Degree

Course Status (compulsory/elective): Elective

Semester (winter/summer): Summer

Language of instruction: English

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

Number of ECTS Allocated: 7

Prerequisites: none

Course Aims:

With the increasing importance of software systems within industry and society, techniques that help in the production of reliable software are becoming increasingly important. The complexity of many software systems requires the application of such systems. Two of the most promising approaches are formal methods and software testing. Traditionally, formal methods and software testing are seen as rivals. Thus, they largely fail to inform each other and there is very little interaction between the two communities.

In recent years consensus was reached, leading to the research directions in which these approaches are complementary. The goal of this course is to introduce and facilitate research of this complementarity.

Learning Outcomes:

- Critically evaluate techniques of software testing
- Critically evaluate the usefulness of formal methods in the testing process.
- Implement formal theories of software testing
- Apply research methods in validation of software

Syllabus:

Theory

Theoretical basis for testing, structural testing, functional testing, the basis for combining formal methods and testing, formal methods based on the model, testing using finite state machines, testing using process algebra, testing using algebraic specification, mutation testing, testing using UML dynamic models, temporal logic and model of checking models and their role in testing and the process of managing software testing.

Practice

-

Required Reading:

1. C. Kaner, J. Falk, H. Q. Nguyen: Testing Computer Software, Wiley, 1999
2. B. Beizer, Software Testing Techniques, International Thomson Press, 1990
3. P. C. Jorgensen, Software Testing: A Craftsman's Approach, second edition, CRC Press, 2004
4. Edmund M. Clarke, Jr., Orna Grumberg and Doron A. Peled, Model Checking, MIT Press, 1999.
5. Ilene Burnstein. Practical Software Testing. Springer-Verlag, 2003
6. Paul Ammann and Jeff Offutt, Introduction to Software Testing, Cambridge University Press, 2008.

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

Teaching Methods:

Lectures are presented using classical methods with a beam projector. Students work on their own to process the topics of research, which they then present and discus their results with the other students and the lecturer.

Knowledge Assessment (maximum of 100 points):

Pre-exam obligations	Points 70	**Final exam**	Points 30

<table>
<tr><td>Active class participation</td><td></td><td>written exam</td><td></td></tr>
<tr><td>Practical work</td><td></td><td>oral exam</td><td></td></tr>
<tr><td>Preliminary exam(s)</td><td></td><td>…….</td><td></td></tr>
<tr><td>Seminar(s)</td><td></td><td></td><td></td></tr>
<tr><td colspan="4">The methods of knowledge assessment may differ; the table presents only some of the options: written exam, oral exam, project presentation, seminars, etc.</td></tr>
</table>