ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
Faculty of Electrical and Computer Engineering
COURSE SYLLABUS
Software Engineering II

1. CODE AND NUMBER OF CREDITS

CODE	FIEC03053	
NUMBER OF CREDITS: 5	**Theoretical: 5**	**Practical: 0**

2. COURSE DESCRIPTION

This course continues with the development of the software project that students initiated during the Software Engineering I course. As part of the project, students should evaluate and monitor each of the project phases tackle during this course: implementation, testing and maintenance. Students should apply standards, best practices and metrics in all of these phases. In addition, students should be part of a project team in order to improve their communication skills as well as their team-working skills.

3. PRE-REQUISITES AND CO-REQUISITES

PRE-REQUISITES	FIEC03046 Software Engineering I
CO-REQUISITES	

4. CORE TEXT AND OTHER REQUIRED REFERENCES FOR THE TEACHING OF THE COURSE

TEXT BOOK	1. Martin Fowler, Refactoring: Improving the Design of Existing Code. 1st Edition, 1999, Addison Wesley. 2. Ian Sommerville, Software Egineering. 9th Edition, 2011, Pearson. 3. Dan Pilone and Russ Miles, Head First Software Development, 2008, O'Reilly Media.
REFERENCES	1. Paul C. Jorgensen, Software Testing. 3rd Edition, 2008, Auerbach Publications. 2. Clemens Szyperski, Component Software. 2nd Edition, 2002, Addison-Wesley. 3. Ela Claridge, Software Testing Module, School of Computer Science, University of Birmingham. 4. George Heineman and William Councill, Component-Based Software Engineering. 1st Edition, 2001, ACM Press Books. 5. Joshua Kerievsky, Refactoring to Patterns. 1st Edition, 2005, Addison Wesley. 6. Eric Freeman and Elisabeth Freeman, Head First Design Patterns. 1st Edition, 2004, O'Reilly Media. 7. William Brown, et al., AntiPatterns: Refactoring Software, Architecture and Project in Crisis. 1st Edition, 1998, John Wiley & Sons, Inc. 8. Oracle University, XML Fundamentals, 10g, Oracle University. 9. Oracle University, Develop Web Services, 10g, Oracle University. 10. ISO/IEC TR 19759:2005, Software Engineering - Guide to the Software Engineering Body of Knowledge. 2005, International Organization for Standardization. 11. Current papers related to the various topics covered in the course.

5. COURSE LEARNING OUTCOMES

At the end of the course, the student will be able to:

1. Describe and use principles for building component based software.
2. Demonstrate the understanding of various technologies and standards for component-based software based on distributed objects.
3. Know the basic concepts related to software maintenance.
4. Describe the main techniques and standards for software testing.
5. Evaluate software testing strategies.
6. Generate appropriate software testing documentation.
7. Design and specify software test cases, and execute software testing procedures for selected problems.
8. Implement and release software application versions that have been properly tested.

9. Evaluate tools and documentation that are used for software maintenance.
10. Use team-working skills.
11. Use oral and written communication skills.

6. COURSE PROGRAM

I. Component Based Software (6 sessions – 15 hours)
- Component-based software engineering
- Software components and layered architectures
- Components characteristics
- Interface description language (IDL)
- XML
- Components composition
- Components and objects
- Components models
- Components frameworks
- Web services
- Middleware
- Cloud computing

II. Configuration Management (1 session – 2.5 hours)
- Configuration management activities
- Terminology

III. Software Testing (2 sessions – 5 hours)
- Software testing fundamentals
- Software testing standards and documentation
- Functional testing and structural testing

IV. Functional Testing (5 sessions – 12.5 hours)
- Introduction
- Methods to define test cases
- Techniques for functional testing

V. Structural Testing (5 sessions – 12.5 hours)
- Introduction
- Techniques for structural testing

VI. Integration Testing (2 sessions – 5 hours)
- Introduction
- Types of integration testing

VII. System Testing (2 sessions – 5 hours)
- Automated testing
- Testing tools
- Metrics and testing evaluation

VIII. Refactoring (3 sessions – 7.5 hours)
- Software design patterns
- Refactoring principles
- Refactoring catalogue
- Case studies

IX. Software Maintenance Management (2 sessions – 5 hours)
- Software maintenance fundamentals
- Service level agreements
- Issues management

7. WORKLOAD: THEORY/PRACTICE

This course is held twice per week. Each of the two class sessions is two hours and a half long. Depending on the topic to be covered classes might have a practical orientation. If a topic includes a practical component, then one of the two weekly sessions will be held in a computer lab of the department.

8. CONTRIBUTION OF THE COURSE TO THE EDUCATION OF THE STUDENT

This course introduces students to component-based software engineering concepts. In addition, during this course students learn how to improve software quality using tools and techniques to formally test the software. Students learn the various types of software testing, and their principles, that are required to develop a software application of varying complexity. Some of these tests are: unit testing, integration testing and system testing. This course also introduces students to configuration management, version management and software maintenance. Students apply knowledge acquired in previous courses that is required during the software development process. Some of these courses are: Web applications development, database systems, data structures and object oriented programming. Students develop skills required for software management, testing and maintenance by finishing the software project they initiated during the Software Engineering I course, where they already executed most of the requirements engineering and software design activities of the project. Students develop team-working and communication skills through the participation in project meetings, meetings with the client, and presentations before the other students. Students have to report the outcomes of the various activities of the project, such as: test plan, coding, and project meetings. Students are committed to developing the Project ethically, respecting any security and confidentiality policy required by the client

BASIC TRAINING	PROFESSIONAL TRAINING	SOCIAL SKILLS DEVELOPMENT
	x	

9. THE RELATIONSHIP BETWEEN THE LEARNING OUTCOMES OF THE COURSE AND THE LEARNING OUTCOMES OF THE DEGREE PROGRAM

LEARNING OUTCOMES OF THE DEGREE PROGRAM	CONTRIBUTION (High, Medium, Low)	LEARNING OUTCOMES OF THE COURSE	THE STUDENT MUST:
a) An ability to apply knowledge of computing and mathematics appropriate to the discipline	-----	-----	-----
b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution	Low	5, 7	Identify and define the interfaces for a component composition problem. Define the test cases to verify the solution to a problem.
c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs	High	1,2,3,6,8	Implement the software Project initiated during the Software Engineering I course. Document the testing process conducted to verify the final release of the developed software.
d) An ability to function effectively on teams to accomplish a common goal	Medium	8,10,11	Be part of a Project team in order to complete the development of a software solution for a real client.
e) An understanding of professional, ethical, legal, security and social issues and responsibilities	Medium	1,2,3,4,5,6,7,8,9,10,11	Respect the principles of academic honesty and the code of ethics of the software engineering profession.
f) An ability to communicate effectively with a range of audiences	High	2,8,10,11	Propose and discuss with the client of the software Project the implementation, testing and deployment of the software solution.
g) An ability to analyze the local and global impact of computing on individuals, organizations, and society	Medium	1,8	Analyze and specify the interfaces and services provided by existent components with the objective of proposing a software solution based on components reuse in order to reduce development times and to better use the human and financial resources of

			the project.
h) Recognition of the need for and an ability to engage in continuing professional development	High	8	Survey some topic related to Software Engineering, and present the results. Demonstrate competences for self-study in order to propose software solutions that consider customer and resources constraints.
i) An ability to use current techniques, skills, and tools necessary for computing practice	Medium	2,4,5,9,10,11	Properly document the tests to the software Project following documentation standards. Students must use software engineering tools and techniques such as: IDE, frameworks, testing tools, middleware.
j) An ability to lead others, manage or undertake projects.	Medium	8,9,10,11	Manage the activities of a software development project for a real client. This project was initiated with the requirements elicitation during the Software Engineering I course.

10. EVALUATION IN THE COURSE

Evaluation activities	
Exams	x
Tests	
Homework/tasks	x
Projects	x
Laboratory/Experiments	
Class participation	x
Visits	
Other	

11. PERSON RESPONSIBLE FOR THE CREATION OF THE SYLLABUS AND THE DATE OF ITS CREATION

Created by	Carlos Monsalve
Date	May 8, 2013

12. APPROVAL

ACADEMIC SECRETARY OF THE ACADEMIC DEPARTMENT	DIRECTOR OF TECHNICAL ACADEMIC SECRETARY
NAME: Sra. Leonor Caicedo G.	NAME: Ing. Marcos Mendoza V.
SIGNATURE: [illegible]	ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL [illegible] Ing. Marcos Mendoza V. DIRECTOR DE LA SECRETARIA TÉCNICA ACADÉMICA
Date of approval by the Directive Council: 2013-334 2013-08-12	

13. VALIDITY OF THE SYLLABUS

RESOLUTION OF THE POLYTECHNIC BOARD:	13-10-269
DATE:	2013-10-17