QFO-AP-FI-MO02	اسم النموذج: Course Syllabus	جامعة فيلادلفيا Philadelphia University
رقم الاصدار : 1 (Revision)	الجهة المصدرة: كلية تكنولوجيا المعلومات	
التاريخ :2017/11/05	الجهة المدققة: عمادة التطوير والجودة	
عدد صفحات النموذج:		

ourse Syllabus

Course Title: Software Reengineering	**Course code:** 0721421
Course Level: 4	**Course prerequisite (s) and/or corequisite (s):** 721420
Lecture Time: 12:10-13:00	**Credit hours: 3**

Academic Staff Specifics

Name	Rank	Office	Office Hours	E-mail Address

Course module description:

It is been observed that 80% of cost spent on software development is spent on maintenance of existing systems. In real life, most software engineers spend their time either helping software systems to continue to meet the needs of their users, or adapting them to meet changing needs. Even new applications are typically constructed by combining existing components and database systems new software.
This module aims to make students aware of the challenges inherent in the reengineering of software systems, and to provide a working understanding of some of the techniques and best practices currently in use for changing software safely.

Course module objectives:

The module aims to

- Understand the concepts and theory related to software reengineering.
- Understand different reengineering techniques used in program understanding and software maintenance
- Learn to use automated reengineering tools in order to adapt software change.

Text book:

Application Software Reengineering
Title: “Application Software Reengineering".
Author(s)/Editor(s): M. Afshar Alam; Tendai Padenga
Publisher: Pearson Education India, 2010

Print ISBN-13: 978-8-131-73185-7

Support materials
Slides and laboratory guide

Teaching methods:
Duration: 15 weeks, 45 hours in total. Based on Lectures,, Tutorials, Problem solving, and demos of using software reengineering tools.

Learning outcomes:

- **Knowledge and understanding**
  1. Locate the factors that make change of existing systems both technically challenging and risky, and the processes required to control change (A2)
  2. Have a knowledge and understanding to recognize the specific problems inherent in the reengineering and evolution of legacy software systems, and be able to apply some of the techniques that can be of use in comprehending and changing them (A2)
  3. Have a knowledge and understanding to recognize the specific problems inherent in the reengineering and evolution of package-based software systems, and be able to apply techniques for designing change-resistant systems from pre-packaged code(A2)
- **Cognitive skills (thinking and analysis)**
  4. Be able to analyze unfamiliar code, in unfamiliar programming language (B1)
  5. Be able to identify and confirm an efficient software reengineering process (B4)
- **Practical skills**
  6. Make appropriate choices regarding the tools and techniques to apply to software evolution problems, trading off costs and limitations against the expected benefits (C2)
  7. Use standard tools and techniques for program comprehension, in order to quickly gain an understanding of an unfamiliar software system (C2)
  8. Use the scientific literature effectively. (C8)
- **Transferable skills**
  9. Be able to communicate effectively with other member of software reengineering project in order to transmit information about founded and repaired bugs(D4)
  10. Solve software reengineering problems (D3)
  11. Communicate effectively with non-specialist as well as computer scientist, (D4)

Assessment instruments

Learning outcomes (1-8) are assessed by examinations; Learning outcomes (4-8) are assessed by tutorials, projects/assignments; Learning outcomes (6-11) are assessed by laboratory works and projects.

Allocation of Marks		
Assessment Instruments	**Date**	**Mark**
First examination		**20 %**
Second examination		**20 %**
Final examination: 50 marks		**40 %**
Quizzes, Short research projects		**20 %**
Total		**100%**

** Make-up exams will be offered for valid reasons only with consent of the Dean. Make-up exams may be different from regular exams in content and format.*

Practical Submissions

The assignments that have work to be assessed will be given to the students in separate documents including the due date and appropriate reading material.

Documentation and Academic Honesty

Documentation style of slides, exam, tutorial, and assignment presentation are available. Academic honesty is based on the principle that one's work is one's own. We encourage all students to accept responsibility for taking academic honesty seriously by being informed, by contributing to a climate in which honesty is valued, and by considering responsible ways to discourage dishonesty in the work of others. Students, faculty, administrators and staff should not condone or tolerate cheating, plagiarism, or falsification, since such activity negatively affects members of the academic community. Plagiarism is the presentation of all or a portion of someone else's work, as one's own, without properly citing/documenting the work.

Plagiarism is unacceptable and will result in a failing grade in the course.

Course/module academic calendar

Week	Basic and support material to be covered	Homework and their due dates	Status
1	The Real Software Lifecycle		
2	Techniques for Corrective Maintenance		
3	Techniques for Preventive Maintenance		
4	Program Comprehension: familiar code reading	Tutorial 1	
5	Program Comprehension: unfamiliar code reading	Tutorial 2	
6 First exam.	Program Comprehension: Tools for program comprehension	Assignment set	
7	Program slicing		
8	Evolution of Legacy Systems	Tutorial 3	
9	Program translation		
10	Restructuring programs	Tutorial 4	
11 Second exam.	Refactoring programs	Tutorial 5	
12	Refactoring programs	Tutorial 6	
13	Reengineering requirements	Assignment due	
14	Reengineering software architecture		
15	Seminar		
16	Final Examination		

On average students need to spend 2 hours of study and preparation for each 50-minute lecture/tutorial.

Attendance policy:

Absence from lectures and/or tutorials shall not exceed 15%. Students who exceed the 15% limit without a medical or emergency excuse acceptable to and approved by the Dean of the relevant college/faculty shall not be allowed to take the final examination and shall receive a mark of zero for the course. If the excuse is approved by the Dean, the student shall be considered to have withdrawn from the course.

Module references

Books

- SOFTWARE EVOLUTION AND MAINTENANCE, A Practitioner's Approach, PRIYADARSHI TRIPATHY KSHIRASAGAR NAIK, John Wiley & Sons, Inc., 2015, (ISBN 978-0-470-60341-3)
- Advances in Software Maintenance Management: Technologies and Solutions, Macario Polo, ed. Idea group publishing. 2003 Secrets of Reverse Engineering, Eldad Eilam, Wiley Publishing, Inc.
- Title: "Reengineering Software: How to Reuse Programming to Build New State-Of-The-Art Software", Roy Rada, Eric Dobby Publishing 1999, ISBN**:** 188899861X

Journals

- Z-E Bouras**.** *Versions of Program Integration*, Handbook of Software Engineering and Knowledge Engineering, Vol. 2, Ed. S.K. Chang, World Scientific Publishing Co., 2002, ISBN: 981-02-4974-8.
- Z-E Bouras. Use cases Evolution. International Journal of Software Engineering, Vol 1, No 1, May 2007.
- Z-E Bouras. Program Translation via a New Dependence Model. International Conference on Software Engineering Research & Practice (SERP'07) Las Vegas, USA, June 25-28, 2007.

Websites

http://www.iam.unibe.ch/~scg/OORP/resources.html
http://www.cs.ualberta.ca/~kenw/toolsdir/related.html
http://proquest.safaribooksonline.com/book/software-engineering-and-development/9788131731857/software-reengineering/149