The University of Jordan
King Abdullah II School for Information Technology
Department of Computer Information Systems

Software Engineering (1902713)
Spring Semester 2015

Instructor: Professor Fawaz AL Zaghoul

Course: Software Engineering (1902713)
Level: Graduate-level
Prerequisite: None.

Course Description

Software processes; component based development; software integration and software reuse. Formal methods in software engineering: formal specification; formal specification languages, Examples of formal specifications. Software cost estimation techniques: algorithmic cost modeling. Quality management and software measurement; Software evolution; software re-engineering; reverse engineering;
Architectural design; distributed systems architectures.Service-oriented software engineering; Aspect-Oriented software engineering.

Objectives

- The main objective of software engineering course is to understand how to produce a high-quality with a cost-effective development of software systems.
- Provide students with a broad knowledge of software engineering processes, methods and tools.
- Provide students with the specific knowledge and skills required to analyze and design complex software systems.
- Prepare students for research in software engineering.

Intended Learning Outcomes (ILOs):

Successful completion of the Software Engineering course should lead to the following outcomes:

A. Knowledge and Under-Standing:

1- Students will be expected to understand the importance of product and process quality in the software development process, through the in-depth understanding of a major quality framework

2- Students should be able to understand how the phases of the lifecycle can be managed using different models of the lifecycle.

3- Students will be expected to appreciate the reasons for a highly structured approach to the software lifecycle and understand the properties of good software and how these relate to different types of software.

4- Students should be able to understand the goals and deliverables of each phase of the software lifecycle, be able to select and apply appropriate techniques to achieving some of these goals and be able to accurately document the results.

B. Intellectual Analytical and Cognitive Skills:

1-Students will be expected to demonstrate how does a software team generate reliable estimates of effort, cost, and project duration.

2- Students should be able to understand and evaluate techniques can be used to formally assess the risks that can have an impact on project success.

3-Evaluate how are black-box and white-box testing methods used to design effective test cases.

4- Identify the basic concepts and principles to the analysis of software requirements.

5- Identify the basic concepts and principles to the design of software activity.

7- Students will be expected to identify important issues in the management of software.

8- Students should be able to understand and evaluate the categorize application domains for computer software.

C. Subject- Specific Skills:

1- Describe how process models can be applied to software development.

2- Students will be expected to have an appreciation of a range of quality management techniques, which enhance product and process quality

3- Students will be expected to have an appreciation of key measures of software quality and productivity and how these might be used within a software quality management framework

D. Transferable Key Skills:

1- Demonstrate how does a software project manager select the set of software engineering work tasks.

2- Evaluate the role of project schedule, quality control and assurance in the development of computer software.

3- Students should be able to identify how is change managed during the development of computer software and after delivery to the customer.

Teaching/Learning Methods

Lectures and Discussions:	**A1-A4; B1**
Homework and Assignments:	**B1-B8; C1-C2.**
Projects:	**D1-D3; C1-C3.**

Course Contents (The main topics)

1- An overview of software processes, requirements engineering and system models.
2- Software design: architectural design, Architectural Styles and Patterns, Distributed systems architecture and application architecture.
3- Software reuse.
4- Component-based software engineering.
5- Formal description.
6- Software testing.
7- Software quality.
8- Project management, quality management, scheduling and cost estimation.
9- Service-oriented software engineering.
10- Aspect-Oriented software engineering.
11- Security engineering.

Methodology

The methodology is based around a lecture on a topic and on a case study, research papers or example which related to the topic.

Research report and presentation

During the course the students (a student or a group) are asked to give a (15- 30) minute presentation of an assigned paper. A discussion of the paper's content and other related topics will follow. Each student is expected to maintain a research report. The report must be submitted at the end of the semester for grading. The report should clearly highlights the major points, and state the contributions (if there is) that the student made.

Homework and Assignments

The students are supposed to prepare a tutorial reports on various topics related to software engineering.

Evaluation

Midterm Exam	30%
Homework and Assignments	15%
Research report and presentation	15%
Final Exam	40%

Main references

- **Ian Sommerville. *Software Engineering*, 9th Edition. Addison Wesley, 2010. http://www.software-engin.com**

- **Roger S. Pressman. Software Engineering A practitioner's Approach, Sixth Edition. McGraw-Hill, 2004.**

References

1. Shari Lawrence Pfleeger, *Software Engineering: Theory and Practice*, 2nd Ed., Prentice-Hall, 2001.
2. J. Rambaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language* Reference Manual. Addison-Wesley, Longman, Mass, USA, 1999.
3. M. Shaw and D. Garlan. Software Architecture: Perspectives on an Emerging Discipline. Prentice Hall Publishing, 1996.
4. D. Schmidt, S. Stal, H. Rohnert, and B. Buschmann. Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects. Vol. 2, John Wiley and Sons, New York, USA, 2000.
5. OMG. *OMG Unified Modeling Language Specification Ver. 1.4.* 2001. Available at http://www.omg.org.
6. J.M. Spivey. The Z Notation: A Reference Manual. Prentice-Hall, 1992.
7. University of Jordan E-library: http://e-library/
8. IEEE Transactions on Software Engineering
9. ACM Transactions on Software Engineering and Methodology
10. Other papers and articles will be made available.