Standard Course Outline for CECS 542: Requirements Engineering (3 Units)

I. General Information

A. Course Number: CECS 542

B. Course Title: "Requirements Engineering"

C. Units: 3

D. Prerequisites: CECS 343 or other basic knowledge about the principles of software engineering and the software lifecycle.

E. Responsible Faculty: Dr. Birgit Penzenstadler, Department of Computer Engineering and Computer Science;

F. Standard course outline prepared by Birgit Penzenstadler, September 13th , 2015

II. Catalog Description

This course aims to equip students to develop techniques of software-intensive systems through successful requirements analysis techniques and requirements engineering. Students learn systematic process of developing requirements through cooperative problem analysis, representation, and validation.
Lecture 2 hours. Semester long team project plus final exam. Letter grade only (A-F).

III. Student Learning Outcomes

After completing the course students will be able to elicit, analyze, document and verify and validate requirements. In particular, they will be able to perform:

- Stakeholder identification and analysis
- Goal identification and analysis
- Creating and refining a system vision
- Developing a domain model of all involved application domains
- Developing a usage model (in the form of UML use cases)
- Eliciting and specifying quality requirements
- Quality assurance techniques
- Requirements management

IV. Curriculum Justification

This course is the essential stepping-stone for conducting successful large, complex software engineering projects. It introduces students in depth to requirements engineering, which lays the foundation for design and all subsequent development phases. It prepares students for complex projects by introducing them to a variety of techniques that enable to analyze and specify requirements from different application domains and stakeholders. The course provides students with the necessary skillset to communicate, analyze, and negotiate with a wide range of potential stakeholders in a project.

V. Outline of Subject Matter

This course exposes students to the problem of determining and specifying what a proposed software system should do, why and for whom the system is needed, not how the system should do it, which is the topic of downstream software engineering activities such as design and coding. There are some nontechnical aspects of the course, with respect to communication and negotiation with multiple stakeholders. Most of the course covers technical approaches to the requirements problem, such as techniques for eliciting stakeholder goals and requirements, notations and models for documenting and specifying requirements, strategies for negotiating requirements, and techniques for analyzing documented requirements. In detail, the course covers the following topics (1 per week):

1. Why do we need Requirements Engineering and what is it?
2. Principles: Definitions, process, roles
3. System Models: Decomposition and abstraction, system views
4. Frameworks: What reference structures can I use for requirements?
5. Business Case Analysis: Why are we building this system?
6. Stakeholders: Who are the people to talk to about requirements?
7. Goals and Constraints: What are the major objectives for the system?
8. System Vision: What exactly do we want to achieve?
9. Domain Models: What are the surrounding systems ours interacts with?
10. Usage Models: How will the system interact with the user?
11. Software quality models: How to determine the quality characteristics?
12. Quality requirements: How to specify which qualities need to be met?
13. Quality assurance: How to ensure that RE is done in a good way?
14. Change management: How to evolve requirements?

VI. Modes of Instruction

The class consists of traditional lectures from faculty and of lab discussion sessions. While there is no official lab time assigned with this class, the assignments that are carried out in teams will be discussed together in class. Students will benefit from structured lectures that cover an adequate number of examples to facilitate student learning and introduce students to the topics covered. The instructor will introduce all requirements engineering methods and techniques tin lectures using a number of examples and hands-on collaborative exercises. Also students will be provided with individual and team assignments and projects that are done outside of class.

VII. Textbook Information

There is no particular textbook assigned to this course. Dr. Penzenstadler detailed the outline presented above in exemplary slide sets and assignments that she will make available to all instructors for reference. Further material can be taken from the books referenced in Section X of this document.

VIII. Instructional Policies Requirements

Instructors may specify their own policies with regard to plagiarism, withdrawal, absences, etc., as long as the policies are consistent with the University policies published in the CSULB Catalog. It is expected that every course will follow University policies on Attendance (PS 01-01), Course Syllabi (PS 04-05), Final Course Grades, Grading Procedures, and Final Assessments (PS 05-07), and Withdrawals (PS 02-02 rev).

IX. Distance Learning/Hybrid Courses

This course will be taught in a traditional format where students attend in-class lectures and carry out assignments individually and in teams.

X. Bibliography

This is a highly selective bibliography to provide instructors with a primary set of resource materials. To ensure brevity, important works may be missing from this list. The list is intended to show the range of materials available to our students.

- Karl Wiegers and Joy Beatty: "Software Requirements"
- Axel van Lamsweerde: "Requirements Engineering"
- Klaus Pohl: "Requirements Engineering"

XI. Student-Level Assessment

The exact set of course assignments will vary depending on the instructor. University policy requires that no single evaluation of student achievement may count for more than one-third of final grade. Appropriate assessment tools include quizzes, exams, written homework, computer-programming assignments and oral presentations.

Sample assignments:

- Eliciting and documenting the stakeholders for a software system.
- Developing a use case in UML.
- Performing a review of quality requirements.

XII. Course-Level Assessment

The exact set of course assignments will vary depending on the instructor. University policy requires that no single evaluation of student achievement may count for more than one-third of final grade. Appropriate assessment tools may include quizzes, exams, written homework, computer-programming assignments and oral presentations. The suggestion is:

1. A semester-long requirements engineering project, composed of individual, written assignments (to practice and demonstrate the skills from the course objectives above).
2. A final examination in form of a written test or an essay.

XIII. Consistency of this Standard Course Outline across Sections

The course coordinator will review this SCO and offer advice and/or materials to each faculty member new to teaching the course. All future syllabi will conform to the SCO. The course coordinator may offer or require regular review of instructors' course materials as well as anonymous samples of student work.