Computer Science 582 - Fall 2016
Software Requirements Analysis and Specification

Instructor:	Bob Lingard
Office:	JD 4401
Office Phone:	677-3825
Email Address:	rlingard@csun.edu
Web URL:	http://www.ecs.csun.edu/~rlingard
Office Hours:	Monday and Wednesday 4:30 – 6:00 PM, and by appointment
Prerequisite:	Comp 380/L (Introduction to Software Engineering) or equivalent
Text:	*Managing Software Requirements: A Use Case Approach*, 2nd Edition, by Dean Leffingwell and Don Widrig
Ticket# 16131	Meets Wednesday 7:00 – 9:45 PM in JD 3510

Course Description

An in-depth study of the early phases of the software development life cycle commonly called software requirements analysis and specification. Topics include the gathering of both functional and nonfunctional requirements, customer communication, requirements prototyping, requirements modeling, requirements validation, the documentation of requirements in terms of a formal software requirements specification, and the management of software requirements.

Course Objectives

Upon successful completion of the course students will be able to:

1. Understand the terminology commonly used in the area of software requirements analysis and specification.
2. Understand the purpose of requirements engineering within the software development lifecycle.
3. Understand the role of requirements engineering within system engineering.
4. Communicate with customers to identify the functional and non-functional requirements for a proposed software system.
5. Develop appropriate use cases and prototypes to clarify software requirements.
6. Identify metrics for assuring the quality of requirement specifications.
7. Use both structured analysis and object-oriented analysis to create models of identified requirements for a software system.
8. Evaluate the quality of software specifications and participate effectively in requirements reviews.
9. Produce as part of a team effort a formal software requirements specification that conforms to IEEE or other recognized standards.
10. Understand the issues related to the ongoing management of established requirements and changes to them during the course of a software development project.

Grading

An absolute grading system will be used, where 90% is required for an A, 80% for a B, 70% for a C, 60% for a D, and anything below 60% is failing. Pluses and minuses will be given for grades within 3 percentage points of the dividing marks. For example, a final total which is 80% or more but less than 83% would correspond to a letter grade of B-, a total of 77% or more but less than 80% would correspond to a C+, etc. The final grade will be based on the following:

Midterm Exam	15%
Final Exam	30%
Group Project	40%
Individual Projects/Homework	15%

Late work will be accepted without penalty only if some compelling reason is provided (preferably in advance) justifying the lateness. Without such a justifiable excuse, late work will be penalized 5% for each **calendar** day that it is late (with a maximum penalty of 50%).

Plagiarism (intentionally or knowingly representing the words, ideas, or work of another as one's own) or any other form of academic dishonesty will not be tolerated. Students who are guilty of such dishonesty will receive no credit for the given assignment or exam and will not be allowed the opportunity to redo the work in question. In addition, incidents of academic dishonesty may be reported to the University and further disciplinary actions are possible.

Exams

The exams may be "take home" exams or "in class" exams. They may be open book or closed book, but, in any case, they must be individual efforts. Discussing the questions on the exam with an individual, other than the instructor, is not permitted.

Group Project

The students in the class will be divided into teams of four to five members each. The group project will be done as a team effort and a single result will be turned in by the team. Members of each team will evaluate the performance of the other team members at the end of the semester. This evaluation will account for 25% of the total Group Project grade (or 10% of the total course grade).

Individual Projects/Assignments

Individual assignments and projects will also be given. Discussion and collaboration with other class members on individual assignments is permitted, and even encouraged, to the extent that said collaboration is a fair and equitable exchange of ideas. That is, one individual should not be doing all the work and sharing it with others. It is permissible to ask other students for help, but it is not permissible to copy the results of others. If several students collectively solve a problem, each should write up the results in his or her own words.

Tentative Schedule by week

1 Introduction to Software Requirements Analysis
The requirements problem
Requirements management
Readings: Chapters 1 - 2

2 Requirements Analysis and the Software Development Process
The software lifecycle
The software team
Understanding the problem and its root causes
Identifying the stakeholders
Defining the solution system boundary
Understanding the constraints on the solution
Readings: Chapters 3 - 5

3 Analyzing the Problem
Business Modeling
System Engineering
Readings: Chapters 6 - 7

4 Understanding User and Stakeholder Needs
Eliciting requirements
System features
Readings: Chapters 8 - 9

5 Techniques for Requirements Elicitation
Interviews
Workshops
Readings: Chapters 10 - 11

6 Techniques for Requirements Elicitation (cont'd)
Brainstorming
Storyboarding
Readings: Chapters 12 -13

7 Introduction to Use Cases
Benefits of a use case approach
Building a use case model
Organizing requirements information
Readings: Chapters 14 -15

8 Mid Term Exam

9 Product Vision and Management
Developing a vision document
Product Management
Readings: Chapter 16 -17

10	Managing Project Scope Establishing scope Managing change Managing the customer Readings: Chapters 18 - 19
11	Developing a Rigorous Set of Requirements Functional requirements Nonfunctional requirements Design constraints Refining the use cases Technical methods Readings: Chapters 20 -21
12	Developing a Rigorous Set of Requirements (cont'd) Supplementary specification Removing ambiguity Technical methods Readings: Chapters 22 -24
13	Assuring System Validity Moving from use cases to design, implementation and testing Readings: Chapters 25 – 26
14	Assuring System Quality Tracing requirements Managing change Assessing requirements quality Readings: Chapters 27 - 29
15	Applying Requirements Analysis Methods Agile methods Process for requirements management Readings: Chapters 30 -31