Computer Science 589 – Fall 2016
Software Metrics

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 & Wednesday 4:30 – 6:00, and by appointment
Prerequisite:	Comp 380/L (Introduction to Software Engineering), Math 340 (Introduction to Probability) or Math 341 (Applied Statistics I), and passing score on the WPE
Text:	*Software Metrics: A Rigorous and Practical Approach,* Third Edition, by Fenton and Bieman
Ticket# 14987:	Meets M 7:00 – 9:45 PM in JD3508

Course Description

The role of metrics and quantitative models in software development. Product metrics, process metrics, measurement models and techniques for empirical validation. Measurement and analysis: implementation of a metrics program. Measuring software size, complexity, and functionality at different stages of software development. Use of measures to predict effort and schedule required for software projects. Measures of software quality. Analyzing defect data to predict software reliability. Performance measures. Management applications for metrics. Tools that support metrics collection, analysis, summary, and presentation.

Course Objectives

Upon successful completion of the course students will be able to

1. Understand the guiding principles and practices for using software metrics to manage software engineering teams and projects.
2. Apply software metrics in specific software engineering environments.
3. Understand the importance of planning, documenting, and implementing a software metrics program.
4. Understand the common problems encountered in software metrics and measurement and how to avoid them in specific projects.
5. Analyze the tradeoffs when selecting quantitative measurement techniques, practices, or tools, and understand how these choices may affect the quality, cost and customer acceptance of software metrics.
6. Discuss current research trends in software metrics

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%
Projects/Homework	30%
Presentations	25%

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.

Projects/Homework

Individual assignments and projects will also be given (some projects may be group projects). 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.

Presentations

Each student will be required to select a technical article on a subject relevant to the course, review and analyze the article, and present a summary and critical evaluation to the class.

Tentative Outline by Week

1. Measurement: What is it?
    a. Why is it important?
    b. Measurement in everyday life
    c. Measurement in software engineering
2. Fundamentals of Measurement Theory
    a. Measurement: What Is It and Why Do It? (Chapter 1, pp. 1-24)
    b. The Basics of Measurement (Chapter 2, pp. 25-51)
3. Measurement Scales and Scale Types (Chapter 3, pp. 51-86)
    a. Types of Measurement Scales
    b. Meaningfulness in Measurement
4. Measuring Software Quality and User Satisfaction
    a. Software Quality Control
    b. Software Defect Removal
    c. Test-Case Coverage
    d. Defect Prevention
    e. Customer-Reported Defects
    f. User Satisfaction
    g. Surveys and interviews
    h. Responsiveness to customer problems
    i. Defect tracking
5. Software Engineering Measurement (Chapter 3, pp. 87-99)

    a. Product metrics
    b. Process metrics
    c. Resource metrics
    d. Determining what to measure
    e. Measuring internal product attributes
    f. Measuring external product attributes
    g. Resource measurement
6. Goal-Question-Metric Paradigm (GQM) (Chapter 3, pp. 100-132)
    a. Determining What to Measure
    b. Applying the Framework
    c. Software Measurement Validation
7. Quality Tools in Software Development
    a. Checklist
    b. Pareto diagram
    c. Histogram
    d. Scatter diagram
    e. Run chart
    f. Control chart
    g. Cause-and-effect diagram
8. Defect Removal Effectiveness
9. Software Models
    a. The Rayleigh Model
    b. The Exponential Model
    c. Reliability growth models
    d. Quality management models
10. Complexity Metrics and Models
    a. Halstead's Software Science
    b. Cyclomatic Complexity
11. Object-Oriented Metrics
12. Conducting Quality Assessments
    a. In-process quality assessments
    b. Software project assessments
13. Using Metrics for Software Process Improvement
14. Developing a Metrics Program
15. Course Review