Wright State University
CORE Scholar

Computer Science & Engineering Syllabi | College of Engineering & Computer Science

Fall 2008

CEG 399: Introduction to Software Testing

John A. Reisner
Wright State University - Main Campus, john.reisner@wright.edu

Follow this and additional works at: https://corescholar.libraries.wright.edu/cecs_syllabi

 Part of the Computer Engineering Commons, and the Computer Sciences Commons

Repository Citation

Reisner, J. A. (2008). CEG 399: Introduction to Software Testing. .
https://corescholar.libraries.wright.edu/cecs_syllabi/100

This Syllabus is brought to you for free and open access by the College of Engineering & Computer Science at CORE Scholar. It has been accepted for inclusion in Computer Science & Engineering Syllabi by an authorized administrator of CORE Scholar. For more information, please contact library-corescholar@wright.edu.

CEG 399: Introduction to Software Testing

Fall Quarter, 2008

Course Description

This course covers software testing strategies, along with established best practices, so students learn how to test their software in a complete and systematic (vice ad-hoc) manner. Particular attention is paid to planning, writing, and executing software testing documentation, i.e., software test plan, to include documented results. Various projects are assigned, designed to illustrate various challenges associated with software testing, and to reinforce the strategies and techniques used to overcome these challenges.

Textbook

Lee Copeland, *A Practitioner's Guide to Software Testing*, Artech House, 2004, ISBN 1-58053-791-X. This is a required textbook for this course.

Reading Assignments

Each week's lessons have corresponding reading assignments. The course lectures are designed to *augment* (not simply *rehash*) these readings.

The course text is a straightforward book. Chapters are written succinctly. It would behoove students to review the material in the book during the week when it is being covered in class.

Course Projects & Lectures

This will be a "learning-by-doing" class. Students will have a series of projects throughout the course, where they will write code, write test plans, execute test plans, and document the results.

Projects will be introduced during class, in the format of interactive discussion exercises. Students should attend class ready to contribute through active participation. No open laptops are allowed in class.

Instructor Contact Info

John Reisner
Office Hours by Appointment
Daytime Phone: 255-3636 x7422 (this is a WPAFB phone number).
email: **john.reisner@wright.edu** (it wouldn't hurt to cc: **john.reisner@afit.edu**)

The instructor is an adjunct faculty member. Most contact will be done via email, phone, or during before- or after-class discussions. Other meetings can be arranged as needed.

Course Objectives

Each student should be able to:

1. Write appropriately comprehensive test plans.
2. Effectively document test plans and results.
3. Develop software using a test-driven approach.
4. Employ effective testing strategies for different needs.
5. Write drivers, stubs, and testware as needed to sufficiently test a program.
6. Verify a program's correctness via a test strategy.

Grading

30% Course Projects

- These consist of a programming project. The emphasis of this project will be testing the software that has been written, with a written test plan.
- All testing is to be performed using a written test plan, which will be developed by the student.
- These are called "weekly" projects; however, in some cases, a project may be extended over two weeks, where students are expected to write the code during the first week, and execute the test plan during the second week. This may happen if the test plans are expected to be exceptionally complex.
- Each project will be graded individually. Although the grade will be based on the thoroughness and quality of the test plan, students are expected to use good programming practices throughout the course.

30% Mid-term Exam

- Mixed-format exam, administered in class.

30% Final Exam

- Comprehensive, mixed-format exam, administered during the school's final exam week.

10% Homework Assignments

- Homework assignments are designed to facilitate deeper comprehension about a lecture topic (in other words, these are "think and respond" assignments).
- There may be up to two assignments per week, but some weeks may have one or zero assignments. Most weeks will not have more than one.
- Homework assignments are different from the weekly projects.
- Answers to these homework assignments generally run about a half to full page in length, and should not take too long to complete.
- Details about these assignments will be found on WebCT.
- Normally, these assignments will be due on Tuesday each week (thus, students have one week to complete a Tuesday assignment and five days to complete a Thursday assignment). Any exceptions to this policy will be mentioned when the homework is assigned.
- Assignments are due at the start of the class/lab session; please have them printed out and ready to turn in at the start of class. If you are unable to attend class, email will be accepted. Emailed assignments should be timestamped before class time (skipping class does not give you a homework extension).
- These assignments will be graded using the SUE grading system (explained on the following page).

Final course grades will be assigned at the instructor's discretion, after all work has been graded, and the grade distribution has been analyzed.

Grading of Course Work

Many of the assignments in this class will be graded subjectively, due to the nature of the work. Many assignments require turn-ins that are not necessarily *right* or *wrong*, but rather well- or poorly-documented, strongly or weakly substantiated, thorough or lax, well-organized or carelessly compiled. Superior work is graded above 90; satisfactory work is graded between 80 and 90, and unsatisfactory work is graded below 80, depending upon the severity of the problems.

Overall, my goal is to assign homework and projects that require much thought, thereby reinforcing understanding and increasing retention.

Course Schedule (possibly subject to change)

Week	Lesson	Date	Lesson Topics	Reading Assignment	Project
1	1	Mon Sep _8_	**Course Introduction** **Terminology & Basics** **Intro to Course Text** **Philosophies & Challenges**	Chapters 1 & 2 Note: Create a $1,000 account at the B&D website. Perform at least two trades.	Project 1. See details on Web CT.
	2	Wed Sep _10_	**Test Cases & Test Plans** **Testable Requirements** **The V-Model & Testing**	Chapters 12 & 14 (skim Chapter 12)	
2	3	Mon Sep _15_	**Black-Box Testing** **Boundary Value Testing** **Equivalence Class Testing**	Section I Introduction Chapters 3 & 4	Project 2. See details on Web CT.
	4	Wed Sep _17_			
3	5	Mon Sep _22_	**Decision Tables** **Orthogonal Arrays**	Chapters 5 & 6	
	6	Wed Sep _24_	**Testing for Robustness:** **Stress Testing, Erroneous Conditions, Pathological Testing**	Outside Readings	
4	7	Mon Sep _29_	**Test-Driven Development** **Use Case Testing**	Chapter 9	Project 3. See details on Web CT.
	8	Wed Oct _1_	**Testing & the Software Lifecycle** **Integration Testing Strategies** **Large System Testing**	Outside Readings	
5	9	Mon Oct _6_	**Domain Analysis Testing**	Chapter 8	
	10	Wed Oct _8_	**Exploratory Testing**	Section III Introduction Chapter 13	
6		Mon Oct _13_	**MIDTERM EXAM**		
	11	Wed Oct _15_	**Intro to White-Box Testing**	Section II Introduction	
7	12	Mon Oct _20_	**Control Flow Testing**	Chapter 10	Project 4. See details on Web CT.
	13	Wed Oct _22_	**Data Flow Testing (Static)**	Chapter 11	
8	14	Mon Oct _27_	**Data Flow Testing (Dynamic)**		
	15	Wed Oct _29_	**TBD**		
9	16	Mon Nov _3_	**Regression Testing**		Project 5. See details on Web CT.
	17	Wed Nov_5_	**Simulation & Testware** **Scalability Problems** **State Transition Testing**	Chapter 7	
10	18	Mon Nov _10_	**Testing Usability** **Performance Testing**		
	19	Wed Nov _12_	**Testing Metrics** **Trends** **How Test Results Shape Testing**	TBD	