4.1 Grading Policy and Syllabus

Exemplary syllabi and grading policies are provided below, in chronological order (2006 through 2011).

Syllabus for CS 617 Requirements Engineering
Spring 2006

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours M 0900 – 0945, W 0900 – 0950 (Robotics (CRMS) Bldg, Room 514D)
or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/Homepage/CS617spr06.htm

Course:	CS 617 Requirements Engineering
Call Number:	06042
Section:	001
Meets:	MWF 10:00 – 10:50
Location:	UK Center for Manufacturing (CRMS) aka Robotics (RMB) Room 323

Description:

The course will examine the requirements phase of the Systems Engineering and Software Engineering lifecycles in detail. Topics will include: requirements elicitation, requirements specification, and requirements analysis. Verification and validation techniques will be emphasized throughout the course. Students will work in small groups to apply requirements engineering techniques.

Course Materials:

Required Text:
L.A. Maciaszek
Requirements Analysis and System Design, 2 ed.
Addison Wesley
ISBN: 0321204646
You must obtain a copy of Maciaszek

Other readings, as assigned: See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 617 will be determined according to these weights:

M.S. students:

Attendance and participation:	5%
Individual assignments:	15%
Team project 1:	15%
Team project 2:	25%
Presentations:	10%
Mid-term:	15%
Final:	15%

Ph.D. students:

Attendance and participation:	5%
Individual assignments:	13%
Team project 1:	15%
Team project 2:	25%
Presentations:	7%
Ph.D. lecture:	5%
Mid-term:	15%
Final:	15%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

There will be a mid-term and a final. The exams will be 50 minute long "in class" exams. The dates for the exams are listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work between groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. Note that the minimum penalty for plagiarism is an F in the course.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex, as well as the CSLab. For information regarding these laboratories, see links under "facilities" from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group projects for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. Proper language usage is required.

Schedule:

Week	**Date**	**Readings**	**Topics**	**Project, Homework, Exam**
1	Wed 1/11/06	Ch. 1	Introduction Software Requirements Specifications	Assign homework 1
1	Fri 1/13/06	Ch. 2	Introduction Software Requirements Specifications	Homework 1 due, assign homework 2
2	Mon 1/16/06	NO CLASS Have fun, be		

		safe!		
2	Wed 1/18/06	Ch. 9	Traceability	Homework 2 due
2	Fri 1/20/06	Ch. 3	Requirements management techniques	
3	Mon 1/23/06	Ch. 3	Requirements management techniques	
3	Wed 1/25/06	Ch. 4	Specification techniques	
3	Fri 1/27/06	Ch. 4	Specification techniques	Assign homework 3
4	Mon 1/30/06		Business Process Models to Requirements	Homework 3 due, assign Homework 4
4	Wed 2/1/06		Business Process Models to Requirements	Ph.D. lecture topic due
4	Fri 2/3/06		Business Process Models to Requirements	Homework 4 due
5	Mon 2/6/06	Requirements Management in Software Processes: Rational Unified Process	Rational Unified Process: Best Practices for Software Development Teams (see URL below) The Ten Essentials of RUP	Assign Homework 5
5	Wed 2/8/06	Requirements Management in Software Processes: Rational Unified Process (RUP)	Rational Unified Process: Best Practices for Software Development Teams The Ten Essentials of RUP	Homework 5 due
5	Fri 2/10/06	Requirements Management in Software Processes: RUP	Rational Unified Process: Best Practices for Software Development Teams The Ten Essentials of RUP	
6	Mon 2/13/06		Requirements Modeling	Hand out project 1
6	Wed 2/19/06		Requirements Modeling	
6	Fri 2/21/06		Requirements Modeling	Ph.D. outline due
7	Mon 2/20/06	Design review Ch 5	Requirements Modeling Case tool demo	
7	Wed	Design review	Requirements Modeling	

	2/22/06	Ch 5	Case tool demo	
7	Fri 2/24/06	Design review Ch 5	Requirements Modeling Case tool demo	
8	Mon 2/27/06		Review for exam	Project 1 draft due
8	Wed 3/1/06	Midterm exam	Midterm exam	Midterm exam
8	Fri 3/3/06			
9	Mon 3/6/06	Ch 4, 5	Specification Models : Use cases	Project 1 Design review
9	Wed 3/8/06	Ch 4, 5	Specification Models : Use cases	
9	Fri 3/10/06	Ch 4, 5	Specification Models : Use cases	
10	Mon 3/13/06 – 3/18/06	NO CLASS Have fun, be safe!		
11	Mon 3/20/06	Ch 5, 6	Specification Models : Class and State Diagrams	Project 1 due, handout Project 2
11	Wed 3/22/06		Project presentations	
11	Fri 3/24/06		Project presentations	
12	Mon 3/27/06	Ch 5, 6	Specification Models : Class and State Diagrams	
12	Wed 3/29/06	Ch 6	Specification Models : Collaboration and Sequence diagrams	
12	Fri 3/31/06	Ch 6	Specification Models : Collaboration and Sequence diagrams	
13	Mon 4/3/06	Formal Specification: A Roadmap Axel van Lamsweerde	Formal Specification techniques	Project 2 draft due
13	Wed 4/5/06	Formal Specification: A Roadmap Axel van Lamsweerde	Formal Specification techniques	

13	Fri 4/7/06	Formal Specification Reading	Formal Specification techniques	
14	Mon 4/10/06	Rolland et al, Guiding goal modeling through scenarios, IEEE Trans. On Software Engineering, 24(12), 1998. Survey Manager system, model	Goal Modeling	Project 2 Design review
14	Wed 4/12/06	Rolland et al, Guiding goal modeling through scenarios, IEEE Trans. On Software Engineering, 24(12), 1998. Survey Manager system, model	Goal Modeling	
14	Fri 4/14/06		Student lectures	
15	Mon 4/17/06		Student lectures	
15	Wed 4/19/06		Requirements Management tools	
15	Fri 4/21/06		Requirements Management tools	
16	Mon 4/24/06		Requirements Specifications to Design	Project 2 due
16	Wed 4/26/06	None	Project Presentations	
16	Fri 4/28/06	None	Project Presentations, Final Exam Review	
Final	Fri 5/5/06 0800 - 1030	Review all readings	Final	Final

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Rational Unified Process: Best Practices for Software Development Teams
http://www.augustana.ab.ca/~mohrj/courses/2000.winter/csc220/papers/rup_best_practices/rup_bestpractices.pdf

The Ten Essentials of RUP http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/dec00/TheTenEssentialsofRUPDec00.pdf

Formal Specification: A Roadmap
Axel van Lamsweerde http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalvanlamsweerde.pdf

Guiding goal modeling using scenarios
Rolland, C.; Souveyet, C.; Achour, C.B.;
Software Engineering, IEEE Transactions on
Volume 24, Issue 12, Dec. 1998 Page(s):1055 – 1071

Barry W. Boehm, Software Engineering, IEEE Trans. On Computers, 25(12):1226-1241, 19.

Parnas, D.L., On criteria to be used in decomposing systems into modules, CACM, vol. 15, no. 12, April ’72, pp.1053-1058. http://portal.acm.org/citation.cfm?id=361623

Wirth, N. Program development by stepwise refinement, CACM, vol. 14, no. 4, 1971, pp. 221-227. http://www.acm.org/classics/dec95/

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.

Syllabus for CS 685 Empirical Software Engineering
Spring 2007

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours MW 1000 – 1050 (Robotics (CRMS) Bldg, Room 514D)
or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/CS685-emp-sw-eng-spring07.htm
Course: CS 685 Empirical Software Engineering
Call Number: 01617
Section: 001
Meets: MWF 11:00 – 11:50
Location: RGAN (Ralph G. Anderson) Room 207

Description:

The course will present the following: Detailed study of the scientific process; particularly using the experimental method. Examination of how empirical studies are carried out in software engineering (by industry and by researchers). Review of the distinction between analytical techniques and empirical techniques. Study of when experimentation is required in software engineering, and what kinds of problems can be solved using experimentation. Examination of how to control variables and to eliminate bias in experimentation. Examination of analysis and presentation of empirical data for decision making. Students will learn how the scientific process should be applied, how and when to apply it in the software engineering area, and how to evaluate empirical evidence. The principles will be reinforced by examination of published experimental studies, and through designing and carrying out small experiments. On completion of the course, students will be in a position to design and carry out experiments in ways appropriate for a given problem, and will acquire skills in analyzing and presenting experimental data.

Course Materials:

Required Text:
Clases Wohlin, Per Runeson, Martin Host, Magnus C. Ohlsson, Bjorn Regnell, Anders Wesslen
Experimentation in Software Engineering: An Introduction
November 1999
Kluwer Academic Pub
ISBN: 0792386825
You must obtain a copy of Wohlin et al.

Other readings, as assigned: See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 685 will be determined according to these weights:

M.S. students:

Attendance and participation:	10%
Paper summaries:	20%
Team research project:	40%
Presentation:	30%

Ph.D. students:

Attendance and participation:	10%
Paper summaries:	20%
Individual research project:	40%
Presentation:	30%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

Papers:

The first nine papers are about experimentation, and the rest are descriptions of experiments. It is important that you read the papers BEFORE the lectures, as the discussion will be very interactive. Turn in simple summaries and evaluations of **four** of the first nine. For **one half** of the remaining papers (10), turn in a short (about one page) summary of the paper. The summaries should: (1) describe the problem in general terms, (2) paraphrase the experimental hypothesis, (3) summarize and critique the design, (4) discuss the conduct of the experiment, (5) explain whether the hypothesis was proved or disproved, and (6) critique the presentation of the paper.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and assignments with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points are good ways to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted **in person** at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work **between** groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. Note that the minimum penalty for plagiarism is an F in the course.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex, as well as the CSLab. For information regarding these laboratories, see links under "facilities" from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group projects for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. Proper language usage is required.

Schedule:

Week	**Date**	**Readings**	**Topics**	**Project, Homework, Exam**
1	Wed 1/10/07	Paper 1	Introduction, Overview of Scientific Method Lecture 1	
1	Fri 1/12/07	Paper 1	Introduction, Overview of Scientific Method, Lecture 1	
2	Mon 1/15/07	NO CLASS Have fun, be safe!		
2	Wed 1/17/07	Papers 2 – 5, Wohlin Chapter 1, 2	Experimentation in Software Engineering, Lecture 2 – 4, Ethics	
2	Fri 1/19/07	Papers 2 – 5, Wohlin Chapter 1, 2	Experimentation in Software Engineering, Lecture 2 – 4, Ethics	
3	Mon 1/22/07	Papers 6 - 9, Wohlin Chapter 3, 4, 5,	Experimentation in Software Engineering, Lecture 5 - 8	Topic selection

		6, 7, 10		
3	Wed 1/24/07	Papers 6 - 9, Wohlin Chapter 3, 4, 5, 6, 7, 10	Experimentation in Software Engineering, Lecture 5 - 8	
3	Fri 1/26/07	Papers 6 - 9, Wohlin Chapter 3, 4, 5, 6, 7, 10	Experimentation in Software Engineering, Lecture 5 - 8	
4	Mon 1/29/07	Papers 10, 11, 12, Wohlin Chapter 8, 9	Metrics and Complexity	
4	Wed 1/31/07	Papers 10, 11, 12, Wohlin Chapter 8, 9	Metrics and Complexity, Guest from the Writing Center	
4	Fri 2/2/07	Papers 10, 11, 12, Wohlin Chapter 8, 9	Metrics and Complexity	
5	Mon 2/5/07	Wohlin Chapter 11, 12	Project Day	Experiment Design Reviews
5	Wed 2/7/07	Wohlin Chapter 11, 12	Project Day	Experiment Design Reviews
5	Fri 2/9/07	Wohlin Chapter 11, 12	Project Day	Experiment Design Reviews
6	Mon 2/12/07	Papers 13, 14, 16	Testing, lecture Assert-Assess	
6	Wed 2/14/07	Papers 13, 14, 16	Testing, lecture Assert-Assess	
6	Fri 2/16/07	Papers 13, 14, 16	Testing, lecture Assert-Assess	
7	Mon 2/19/07	Papers 19, 21, 22	Maintenance, lecture Writing	
7	Wed 2/21/07	Papers 19, 21, 22	Maintenance, lecture Writing	
7	Fri 2/23/07	Papers 19, 21, 22	Maintenance, lecture Writing	
8	Mon 2/26/07	Papers 32, 33, 34	Traceability	
8	Wed 2/28/07	Papers 32, 33, 34	Traceability	
8	Fri 3/2/07	Papers 32, 33, 34	Traceability	
9	Mon	Papers 23, 24	Requirements & Design	Hand out sample

	3/5/07			paper
9	Wed 3/7/07	Papers 23, 24	Requirements & Design	
9	Fri 3/9/07	Papers 23, 24	Requirements & Design	
10	Mon 3/12/07 – 3/16/07	NO CLASS Have fun, be safe!		
11	Mon 3/19/07	Papers 25, 26	Design	Draft paper due
11	Wed 3/21/07	Papers 25, 26	Design	
11	Fri 3/23/07	Papers 25, 26	Design	
12	Mon 3/26/07	Papers 27, 28	Design, Lecture Presentations	Reviews due
12	Wed 3/28/07	Papers 27, 28	Design, Lecture Presentations	
12	Fri 3/30/07	Papers 27, 28	Design, Lecture Presentations	
13	Mon 4/2/07	Papers 29, 30	HCI, Management and Inspections	Hand out sample presentation
13	Wed 4/4/07	Papers 29, 30	HCI, Management and Inspections	
13	Fri 4/6/07	Papers 29, 30	HCI, Management and Inspections	
14	Mon 4/9/07	None	Project Presentations	Final research papers due
14	Wed 4/11/07	None	Project Presentations	
14	Fri 4/13/07	None	Project Presentations	
15	Mon 4/16/07	None	Project Presentations	
15	Wed 4/18/07	None	Project Presentations	
15	Fri 4/20/07	None	Project Presentations	
16	Mon 4/23/07	None	Project Presentations	
16	Wed 4/25/07	None	Project Presentations	
16	Fri 4/27/07	None	Project Presentations	
Final	Wed	None	Project Presentations – if	

	5/2/07 1030 - 1300		time slot needed	

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Readings:

Empirical Methods Overview

1. **National Research Council**, *Academic Careers for Experimental Computer Scientists and Engineers, Ch. 1*, National Acadamy Press, pages 9-33, 1994. TOC PS
2. **Fenton**, Norman, Shari Lawrence Pfleeger and Robert L. Glass, "Science and Substance: A Challenge to Software Engineers", *IEEE Software*, V. 11, N. 4, pages 86-95, July 1994. PDF (not working) HTML (works, but ugly)
3. **Tichy**, Walter F., "Hints for Reviewing Empirical Work in Software Engineering", *Empirical Software Engineering*, 5(4):309-312, December 2000. EMSE Home
4. **Amschler Andrews,** Anneliese and Arundeep S. Pradhan, "Ethical Issues in Empirical Software Engineering: The Limits of Policy", *Empirical Software Engineering*, 6(2):105-110, June 2001. EMSE Home
5. **Zendler**, Andreas, "A Preliminary Software Engineering Theory as Investigated by Published Experiments", *Empirical Software Engineering*, 6(2):161-180, June 2001. EMSE Home
6. **Harrison**, Warren "Editorial: Open Source and Empirical Software Engineering", *Empirical Software Engineering*, 6(3):193-194, September 2001. EMSE Home
7. **Shull**, Forrest, Manoel G. Mendoncça, Victor Basili, et al. "Knowledge-Sharing Issues in Experimental Software Engineering", *Empirical Software Engineering*, (9)1-2:111-137, March 2004. EMSE Home
8. **Karahasanovic',** Amela, Bente Anda, Erik Arisholm, Siw Elisabeth Hove, Magne Jørgensen, Dag I K Sjøberg and Ray Welland, "Collecting Feedback During Software Engineering Experiments", *Empirical Software Engineering*, 10(2):113-147, April 2005. EMSE Home
9. **Offutt**, Jeff, Yuan Yang and Jane Hayes, "SEEWeb: Making Experimental Artifacts Available", *Workshop on Empirical Research in Software Testing*, Boston, MA, July 2004. PDF

Metrics and Complexity

10. • **L. Briand** and J. Wust, "Empirical Studies of Quality Models in Object-Oriented Systems", *Advances in Computers*, vol. 56, 2002, Academic Press. Briand's homepage
11. • **Fenton**, Norman and Niclas Ohlsson, "Quantitative Analysis of Faults and Failures in a Complex Software System", *IEEE Transactions on Software Engineering*, (26)8:797-814, August 2000. PDF
12. • **Wohlin**, Claes, and Anneliese Amschler Andrews "Prioritizing and Assessing Software Project Success Factors and Project Characteristics using Subjective Data", *Empirical Software Engineering*, (8)3:285-308, September 2003. EMSE Home

Testing

13. • **Juristo**, Natalia, Ana M. Moreno, Sira Vegas "Reviewing 25 Years of Testing Technique Experiments", *Empirical Software Engineering*, (9)1-2:7-44, March 2004. EMSE Home
14. • **Ma**, Yu-Seung, Jeff Offutt and Yong Rae Kwon, "MuJava: An Automated Class Mutation System", *Journal of Software Testing, Verification and Reliability*, 15(2):97-133, June 2005. PDF local copy
15. • **Roger T. Alexander** and Jeff Offutt, "Empirical Evaluation of Coupling-based Testing Techniques for Object-oriented Programs", submitted. PDF
16. • **Lionel C. Briand**, Massimiliano Di Penta and Yvan Labiche, "Assessing and Improving State-Based Class Testing: A Series of Experiments", *IEEE Transactions on Software Engineering*, 30(11), November 2004. PDF
17. • **Grindal**, Mats, Jeff Offutt and Jonas Mellin, "State-of-Practice: An Investigation of Testing Maturity", submitted. Preliminary version
18. • ~~**Stuart C. Reid**, "An Empirical Analysis of Equivalence Partitioning, Boundary Value Analysis and Random Testing", Proceedings of the 4th International Software Metrics Symposium (METRICS '97), 1997. PDF~~

Maintenance

19. • **Kajko-Mattsson**, Mira, "A Survey of Documentation Practice within Corrective Maintenance", *Empirical Software Engineering*, 10(1):31-55, January 2005. EMSE Home
20. • ~~**Liguo Yu**, Stephen R. Schach, Kai Chen and Jeff Offutt , "Categorization of Common Coupling and its Application to the Maintainability of the Linux Kernel", *IEEE Transactions on Software Engineering*, 30(10):694-706, October 2004. PDF local~~
21. • **Kai Chen**, Stephen R. Schach, Liguo Yu, Jeff Offutt and Gillian Z. Heller, "Open-Source Change Logs", *Kluwer's Empirical Software Engineering*, 9(3):197-210, September 2004. online EMSE Home
22. • **Stephen R. Schach**, Bo Jin, Liguo Yu, Gillian Z. Heller and Jeff Offutt, "Determining the Distribution of Maintenance Categories: Survey versus Measurement", *Kluwer's Empirical Software Engineering*, 8(4):351-365, December 2003. online EMSE Home

Requirements

23. • **Damian**, Daniela, James Chisan, Lakshminarayanan Vaidyanathasamy and Yogendra Pal, "Requirements Engineering and Downstream Software Development: Findings from a Case Study", *Empirical Software Engineering*, (10)3:255-28, July 2005. EMSE Home

Design

24. • **Iris Reinhartz-Berger** and Dov Dori, "OPM vs. UML--Experimenting with Comprehension and Construction of Web Application Models", *Empirical Software Engineering*, 10(1), January 2005. EMSE Home
25. • **Marek Vokáccaron**, Walter Tichy, Dag I. K. SjØberg, Erik Arisholm and Magne Aldrin, "A Controlled Experiment Comparing the Maintainability of Programs Designed with and without Design Patterns-A Replication in a Real Programming Environment", *Empirical Software Engineering*, 9(3):149-195, September 2004. EMSE Home
26. • **Anda**, Bente and Dag I. K. Sjøberg, "Investigating the Role of Use Cases in the Construction of Class Diagrams", *Empirical Software Engineering*, (10)3:285-309, July 2005. EMSE Home

27. • **Svahnberg**, Mikael and Claes Wohlin "An Investigation of a Method for Identifying a Software Architecture Candidate with Respect to Quality Attributes", *Empirical Software Engineering*, (10)2:149-181, April 2005. EMSE Home
28. • **Knight**, John C. and Nancy G. Leveson, "An Experimental Evaluation of the Assumption of Independence in Multiversion Programming", *IEEE Transactions on Software Engineering*, (SE-12)1:96-109, January 1986. NEC Research Index (CiteSeer)

HCI

29. • **Miara**, Richard J., Joyce A. Musselman, Juan A. Navarro, and Ben Shneiderman, "Program Indentation and Comprehensibility", *Communications of the ACM*, (26)11:861-867, November 1983. ACM

Management and Inspections

30. • **McDonald**, James, "The Impact of Project Planning Team Experience on Software Project Cost Estimates", *Empirical Software Engineering*, (10)2:219-234, April 2005. EMSE Home
31. • ~~**Thelin**, Thomas, Per Runeson, Claes Wohlin, et al. "Evaluation of Usage-Based Reading-Conclusions after Three Experiments", *Empirical Software Engineering*, (9)1-2:77-110, March 2004. EMSE Home~~

Traceability

32. • O.C.Z. Gotel and A.C.W. Finkelstein. An analysis of the requirements traceability problem. In 1st International Conference on Requirements Engineering, pages 94--101, 1994. PDF
33. Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., and Merlo, E. Recovering Traceability Links between Code and Documentation. IEEE Transactions on Software Engineering, Volume 28, No. 10, October 2002, 970-983. PDF
34. Jane Huffman Hayes, Alex Dekhtyar: A Framework for Comparing Requirements Tracing Experiments. International Journal of Software Engineering and Knowledge Engineering 15(5): 751-782 (2005) PDF

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.

Syllabus for CS 585-001 Software Testing and Quality Evaluation

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building 257-3171
Office hours 1:00 – 1:50 pm MW or by appointment (Robotics Room 514D)

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/ CS485-Fall08.htm

Course: CS 485 Software Testing

Section:	001
Meets:	MWF 2:00 – 2:50 pm
Location:	Oliver H Raymond Building (OHR) Room C226

Description:

Concepts and techniques for testing software and assuring its quality. Topics cover software testing at the unit, module, subsystem, and system levels, automatic and manual techniques for generating and validating test data, the testing process, static vs. dynamic analysis, functional testing, inspections, and reliability assessment. *Professor's note*: The course will attempt to prepare students to test software in structured, organized ways. This course should provide practical knowledge of a variety of ways to test software, an understanding of some of the tradeoffs between testing techniques, and a feel for the practice of software testing **and** the research in software testing.

Course outcomes:

The student shall know the levels of testing and be able to define them.
The student shall know the definitions of key testing terms such as coverage criterion and subsumption.
The student shall understand how to generate test cases that achieve data flow coverage and/or determine if given test cases achieve data flow coverage for a given program.
The student shall understand how to generate logic expression coverage test cases and/or determine if given test cases achieve logic expression coverage.
The student shall understand Syntax-Based Coverage Criteria.
The student shall understand class integration test order.

Course Materials:

Required Texts:

- Paul Ammann and Jeff Offutt, *Introduction to Software Testing*, Cambridge University Press, Cambridge, UK, ISBN 0-52188-038-1, 2008.
- Various artifacts available over the web. Details as needed.

Other readings, as assigned: See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 485 will be determined according to these weights:

Undergraduate students:

Weekly homework assignments/paper summaries: 45%

Weekly quizzes:	45%
Final:	10%

Graduate students:

Weekly homework assignments/paper summaries:	32%
Additional paper summaries:	11%
Weekly quizzes:	32%
Short paper and/or presentation:	15%
Final:	10%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

The lowest two homework and quiz grades will be dropped; allowing for missed classes. That is, **two total grades** will be dropped from the total set of homework grades and quiz grades, not four total.

There will be one closed book, in-class, comprehensive final. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class has started are **late**.

In recognition of the fact that students have occasional but unavoidable commitments that preclude attendance at every class, I drop each student's two lowest scores prior to final grade computations. I select the combination of homework and quiz scores most advantageous to the student. For example, I may drop one quiz score and one homework, or two homework assignments, or any other combination totaling two.

In view of this policy, **late homework is not accepted and there are no make-up quizzes. Please do not ask for an exception.**

Attendance/Absences:

Arrival after class has started is disruptive to the class and is not appreciated. Arrival after student presentations have begun will not be allowed. Arrival after the homework assignments have been collected constitutes a 0 for that assignment. Absence the day of a quiz, test or presentation constitutes a 0 for that grade element. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student’s responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work between groups will be acceptable. The University of Kentucky’s guidelines regarding academic dishonesty will be strictly enforced. Note that the minimum penalty for plagiarism is an F in the course.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department, located in Room 203 of the Engineering Annex as well as the CSLab. For information regarding these laboratories, see links under "facilities" from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Paper Summaries [3]:

You are required to read and evaluate each of the assigned readings prior to discussion in class. Summaries/evaluations are due at the start of class. Each paper summary should be on a separate sheet of paper, and include:

· The title, and first author's name
· The main point that the article seemed to make (2-5 sentences)
· Two subjective numerical ratings on a 1-to-6 scale (1 low, 6 high):
a) How important is the material covered in the article?
b) How well-written was the article?
· Two to three paragraphs concerning the content of the article, containing either:
a) A question about the article, such as one that you or someone reading the paper for the first time might have to stop and study, look elsewhere, or re-read to find an answer. Questions should be accompanied by an elaboration of the question, and/or a discussion of its relevance.
b) A comment on the article, such as discussion of application, classification, comparison, and/or evaluation of method or methods.
c) What you liked, disliked, found interesting or found unclear in the article.

Also, see "How to Read an Engineering Research Paper" by Bill Griswold for additional ideas (http://www-cse.ucsd.edu/users/wgg/CSE210/howtoread.html).

Paper evaluations will be graded according to the following scale: 0: not submitted, 1: marginal, 2: what was expected, 3: outstanding. You are expected to have read all articles. Proper language usage is required.

Homework assignments:

There will be small computer and homework assignments for most of our topics and/or paper summaries (see above). Some will require pencil and paper and others will require modest programming or use of tools available from the Web. Proper language usage is required.

Quizzes:

We will have weekly (roughly) quizzes and no midterm exam. Quizzes will be given during the **first 10 minutes** of each class and no makeup or late quizzes will be given.

Short Paper and/or Presentation:

Graduate students will prepare a 3 to 4 page paper that describes a current testing topic, of the student's choosing. Proper language usage is required. If time permits, students will make a short presentation to the class on the topic.

Schedule:

Week	Date	Readings	Topics	Assignments, Exam
1	Wed 8/27/08	Ammann/Offutt Chapter 1	Introduction and Background on Coverage Based Testing	
1	Fri 8/29/08	Ammann/Offutt Chapter 1, (Read Ariane 5 Flight Failure– graduate students)	Introduction and Background	Homework 1, Graduate student paper summary
2	Mon 9/1/08	NO CLASS	NO CLASS	NO CLASS
2	Wed 9/3/08	AO 2.1-2.2	Graph Coverage	Quiz 1
2	Fri 9/5/08	AO 2.1-2.2	Graph Coverage	
3	Mon 9/8/08	AO 2.1-2.2	Graph Coverage	Homework 2
3	Wed 9/10/08	AO 2.3, (Read TBD paper – graduate students)	Graph Coverage for code	Graduate student paper summary
3	Fri 9/12/08	AO 2.3	Graph Coverage for code, Code of ethics	Quiz 2
4	Mon 9/15/08	AO 2.3, AO 2.4	Graph Coverage for code	Homework 3
4	Wed 9/17/08	AO 2.4 -2.5	Graph Coverage for other artifacts	Quiz 3
4	Fri 9/19/08	AO 2.4-2.5	Graph Coverage for other artifacts	Topics for graduate papers due
5	Mon 9/22/08	AO 2.4-2.5	Graph Coverage for other artifacts	Homework 4

5	Wed 9/24/08	AO 2.6	Use Case graph coverage	Quiz 4
5	Fri 9/26/08	AO 2.6	Use Case graph coverage	2
6	Mon 9/29/08	AO 2.6, 3.1 and 3.2	Use Case graph coverage	Homework 5
6	Wed 10/1/08	AO 3.1-3.2	Logic testing	Quiz 5
6	Fri 10/3/08	AO 3.1-3.2	Logic testing	
7	Mon 10/6/08	AO 3.1-3.2, 3.3	Logic testing	Homework 6
7	Wed 10/8/08	AO 3.3-3.4	More logic testing	Quiz 6
7	Fri 10/10/08	AO 3.3-3.4	More logic testing	Homework 7
8	Mon 10/13/08	AO 3.5	Still more logic testing	Quiz 7
8	Wed 10/15/08	AO 3.5, 4	Still more logic testing	Homework 8
8	Fri 10/17/08	AO 4	Input space partitioning	
9	Mon 10/20/08	AO 4	Input space partitioning	3 Homework 9
9	Wed 10/22/08	AO 3.6	Disjunctive normal form testing	Quiz 8
9	Fri 10/24/08	AO 3.6	Disjunctive normal form testing	Draft graduate papers due
10	Mon 10/27/08	AO 5.1	Syntax testing/mutation analysis	Homework 10
10	Wed 10/29/08	AO 5.1	Syntax testing/mutation analysis	Quiz 9
10	Fri 10/31/08	AO 5.2	Syntax testing/mutation analysis	Homework 11
11	Mon 11/3/08	AO 5.2	Syntax testing/mutation analysis	

11	Wed 11/5/08	4 Read Hutchins et al paper	5 Dataflow and Controlflow-based test adequacy criteria	Quiz 10, Paper Summary
11	Fri 11/7/08	AO 5.3	Integration and OO	
12	Mon 11/10/08	AO 5.3	Integration and OO	Quiz 11
12	Wed 11/12/08	AO 5.3, (Read TBD paper – graduate students)	Integration and OO	Graduate student paper summary
12	Fri 11/14/08	AO 5.3	Integration and OO	
13	Mon 11/17/08	AO 5.5	input space grammars	Quiz 12, Homework 12
13	Wed 11/19/08	AO 5.5	Input space grammars	
13	Fri 11/21/08	6 Read Frankl and Weiss paper	7 All-uses	Paper Summary
14	Mon 11/24/08	8 AO 6.4	9 Test Plans	10 Quiz 13, Graduate papers due
14	Wed 11/24/08	NO CLASS	NO CLASS	NO CLASS
14	Fri 11/26/08	NO CLASS	NO CLASS	NO CLASS

15	Mon 12/1/08	11	AO 7.1	12	Testing OO software	
15	Wed 12/3/08	13	AO 7.1	14	Testing OO software	Quiz 14
15	Fri 12/5/08	15	AO 7.1	16	Testing OO software	
16	Mon 12/8/08	17	Read Offutt paper	18	Software testing coupling	Quiz 15, Paper Summary
16	Wed 12/10/08	19	Catch up day (if needed)	20		
16	Fri 12/12/08			Course Wrap up, Final Exam Review		
Final	Fri 12/19/08 0800-1030	**Review all readings, assignments**				**Exam (Final)**

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Report by the Inquiry Board. ARIANE 5 Flight 501 Failure. Technical report, European Space Agency, 1996. J.L. Lions, Chairman of the Board.
http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html
Experiments of the effectiveness of dataflow- and controlflow-based test adequacy criteria
International Conference on Software Engineering
Proceedings of the 16th international conference on Software engineering
Sorrento, Italy, Pages: 191 – 200, Year of Publication: 1994, ISBN:0-8186-5855-X
Authors**:** Monica Hutchins, Herb Foster, Tarak Goradia, Thomas Ostrand
Paper available from our course web page

An experimental comparison of the effectiveness of the all-uses and all-edges adequacy criteria International Symposium on Software Testing and Analysis
Proceedings of the symposium on Testing, analysis, and verification, Victoria, British Columbia, Canada

Pages: 154 – 164, Year of Publication: 1991, ISBN:0-89791-449-X
Authors: Phyllis G. Frankl, Stewart N. Weiss
Paper available from our course web page

21 Investigations of the software testing coupling effect
ACM Transactions on Software Engineering and Methodology (TOSEM)
Volume 1 , Issue 1 (January 1992), Pages: 5 – 20, Year of Publication: 1992
ISSN:1049-331X
Author: A. Jefferson Offutt
Paper available from our course web page

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.
[3] http://www.cs.ucsc.edu/~ejw/courses/290gw02/assignments.htm

Syllabus for CS 499-001 Senior Design Project
Spring 2009

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours TR 0915 – 1000 (Robotics (CRMS) Bldg, Room 514D) or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/CS499spring09.htm

Course:	CS 499 Senior Design Project
Section:	001
Meets:	TR 0800 - 0915
Location:	Anderson Hall (FPAT) Room 263

Description:

This is a project course. Students will work in small groups to design and implement systems of current interest to computer scientists. The course will also provide a high-level overview of the software engineering discipline: software requirements, software design, software construction, software management, and software quality.

Course Outcomes:

- The student shall know the phases of the software development lifecycle and be able to define them. (C1)
- The student shall know the difference between project and process metrics. (C2)

- The student shall be able to define the terms version control and change control. (C3)
- The student shall be familiar with methods for performing requirements elicitation and requirements analysis. (C4)
- The student shall be able to discuss important design principles such as information hiding and abstraction. (C5)
- The student shall be able to discuss the differences between structured and object oriented analysis and design. (C6)
- The student shall be able to define key testing terms such as black box testing and white box testing. (C7)
- The student shall be able to perform the activities of the software lifecycle for a small to medium software project. (C8)

Course Materials:

Required Text:
Roger S. Pressman
Software Engineering: A Practitioner's Approach, *** *Fifth Edition****
McGraw-Hill
ISBN: 0-07-052182-4
You must obtain a copy of Pressman

Recommended Texts:

Frederick P. Brooks, *Mythical Man Month, 2nd Edition*, Addison Wesley
ISBN: 0-201-83595-9

You do not have to obtain these, though you may choose to. Also, copies have been placed on reserve in the Engineering Library (3rd floor Anderson Hall)

Other readings, as assigned:

These are available via hyperlink in this syllabus or are on our course web page. See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 499 will be determined according to these weights:

Attendance and participation*:	10%
Presentations:	25%
Projects:	50%
Mid-term:	15%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

*Includes preparation of resume and completion of senior survey.

There will be a mid-term. The exam will be a 50 minute long “in class” exam. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student’s responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work between groups will be acceptable. The University of Kentucky’s guidelines regarding academic dishonesty will be strictly enforced. Note that the minimum penalty for plagiarism is an E in the course.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex, as well as the CSLab. For information regarding these laboratories, see links under “facilities” from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group project for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. **Proper language usage is required.**

Schedule:

Week	**Date**	**Readings**	**Topics**	**Project, Exam**
1	Thu 1/15/09	Pressman Ch. 1,2	Product, Process	
2	Tue 1/20/09	Pressman Ch. 3, Boehm S/W Eng. paper	Project Management	
2	Thu 1/22/09	Pressman Ch. 4,5	Metrics, Project Planning	
3	Tue 1/27/09	Pressman Ch. 10	Systems Engineering	
3	Thu 1/29/09	No class		
4	Tue 2/3/09	Pressman Ch. 11	Analysis Concepts	**Resumes due**
4	Thu 2/5/09	Pressman Ch. 12	Analysis Modeling	
5	Tue 2/10/09	Pressman Ch. 6,8	Risk, SQA	Hand out project, start Phase I
5	Thu 2/12/09	Pressman Ch. 7,9	Project Scheduling, SCM	
6	Tue 2/17/09	No class		
6	Thu 2/19/09	Pressman Ch. 13, Parnas paper	Design Concepts	
7	Tue 2/24/09	Pressman Ch. 14, Wirth paper	Architecture Design	
7	Thu 2/26/09	Pressman Ch. 15,16	User Interface Design, Other Design Topics	
8	Tue 3/3/09	None	Project Presentations	**Project Phase I due**, Start Phase II
8	Thu 3/5/09	Pressman Ch. 19,20	Technical Metrics, OO Concepts	
9	Tue 3/10/09	No class		

9	Thu 3/12/09	Pressman Ch. 21,22	OOA, OOD	
10	3/16-3/21	No class – SPRING BREAK. Be careful, have fun		
11	Tue 3/24/09	Pressman Ch. 17, 18	Software Testing Techniques and Strategies, Exam review	
11	Thu 3/26/09	**Midterm exam**	**Midterm exam**	**Midterm exam**
12	Tue 3/31/09	Pressman Ch. 18, 24, Chidamber paper	Software Testing Techniques and Strategies , OO Metrics	
12	Thu 4/2/09	Maintenance readings, Brooks Ch. 16	Software maintenance	
13	Tue 4/7/09	None	Project Presentations	**Project Phase II due,** Start Phase III
13	Thu 4/9/09	None	Project Presentations	
14	Tue 4/14/09	No class		
14	Thu 4/16/09	Experimental software eng. Readings, Brooks Ch. 17	Experimental software engineering	
15	Tue 4/21/09	Software reliability reading	Software reliability	**Senior surveys due**
15	Thu 4/23/09	Software reliability reading	Software reliability	
16	Tue 4/28/09	None	Project Presentations	**Completed Project due**
16	Thu 4/30/09	None	Project Presentations	

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Barry W. Boehm, Software Engineering, IEEE Trans. On Computers, 25(12):1226-1241, 19. – see course web page

Boehm, B. A Spiral Model for Software Development and Enhancement, Computer, Vol. 21, no. 5, May '88, pp. 61-72. - see course web page

Parnas, D.L., On criteria to be used in decomposing systems into modules, CACM, vol. 15, no. 12, April '72, pp.1053-1058. http://portal.acm.org/citation.cfm?id=361623

Wirth, N. Program development by stepwise refinement, CACM, vol. 14, no. 4, 1971, pp. 221-227. http://www.acm.org/classics/dec95/

Musa, J.D., and Ackerman, A.F., Quantifying software validation: when to stop testing? IEEE SW, May 1989, pp. 19-27. - see course web page

Chidamber, S.R. and C.F. Kemerer, A metrics suite for object-oriented design, IEEE TSE, vol. SE-20, no. 6, June '94, pp.476-493. http://portal.acm.org/citation.cfm?id=631131

Frakes, W.B. and T.P. Pole, An empirical study of representation methods for reusable software components, IEEE TSE, vol SE-20, no. 8, Aug '94, pp. 617-630. - see course web page

Kiczales, G., Lamping, J. , Mendhekar, A., Maeda, C., Lopes, C.V., Loingtier, J.-M., and Irwin, J. Aspect--Oriented Programming. In European Conference on Object--Oriented Programming, ECOOP'97,
LNCS 1241, pages 220--242, Finland, June 1997. Springer--Verlag.
http://www2.parc.com/csl/groups/sda/publications/papers/Kiczales-ECOOP97/for-web.pdf

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.

Syllabus for CS 616-001 Software Engineering
Spring 2009

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building 257-3171
Office hours 9:15 – 10:00 TR (Robotics 514D) **or by appointment**

Course information:

Course homepage
http://selab.netlab.uky.edu/homepage/CS616spring09.htm

Course:	CS 616 Software Engineering
Section:	001
Meets:	TR 11:00 – 12:15

Location: RGAN Room 202

Description:

This course provides an overview of the software engineering discipline: software requirements, software design, software construction, software management, and software quality. Traceability, testing, and validation techniques will be emphasized throughout the course. Programs and program fragments will be developed and studied throughout the course to illustrate specific problems encountered in the lifecycle development of software systems.

Course Outcomes:

- The student shall know the phases of the software development lifecycle and be able to define them. (CO1)
- The student shall know the definitions of key software engineering terms such as verification, validation, and formal technical review. (CO2)
- The student shall be familiar with a number of software process models such as waterfall and RAD. (CO3)
- The student shall be familiar with key software project management concepts such as communication. (CO4)
- The student shall know the difference between project and process metrics. (CO5)
- The student shall be able to list the key activities of software quality assurance. (CO6)
- The student shall be able to define the terms version control and change control. (CO7)
- The student shall be familiar with methods for performing requirements elicitation and requirements analysis. (CO8)
- The student shall be able to discuss important design principles such as information hiding and abstraction. (CO9)
- The student shall be able to discuss the differences between structured and object oriented analysis and design. (C10)
- The student shall be able to define key testing terms such as black box testing and white box testing. (C11)
- The student shall be able to perform the activities of the software lifecycle for a small to medium software project. (C12)

Course Materials:

Required Text:
Roger S. Pressman
Software Engineering: A Practitioner's Approach, ****Fifth Edition ****
McGraw-Hill
ISBN: 0-07-052182-4
You must obtain a copy of Pressman

Recommended Texts:

Frederick P. Brooks, *Mythical Man Month, 2nd Edition*, Addison Wesley
ISBN: 0-201-83595-9

You do not have to obtain these, though you may choose to. Also, copies have been placed on reserve in the Engineering Library (3rd floor Anderson Hall)

Other readings, as assigned:

these are available on-line (links embedded in this document) or via the course web page. See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 616 will be determined according to these weights:

Master's Students:

Attendance and participation:	10%
Presentations:	20%
Projects:	37%
Final:	33%

Ph.D. Students:

Attendance and participation:	10%
Presentations:	13%
Projects:	36%
Lecture:	10%
Final:	31%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

There will be a final. The exam will be an "in class" exam. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Attendance/Absences:

Arrival after class has started is disruptive to the class and is not appreciated. **Arrival after team presentations or PhD lectures have begun will not be allowed. Please wait in the hall until the team is done, then come in and take your seat.** Arrival after the quiz has commenced constitutes a 0 for the quiz. Absence the day of a quiz, test, or presentation constitutes a 0 for that grade element. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work between groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. Note that the minimum penalty for plagiarism is an E in the course.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department, located in Room 203 of the Engineering Annex as well as the CSLab. For information regarding these laboratories, see links under “facilities” from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group project for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. **Proper language usage is required.**

Schedule:

Week	Date	Readings	Topics	Project, Exam
1	Thu 1/15/09	Pressman Ch. 1,2	Product, Process	
2	Tue 1/20/09	Pressman Ch. 3, Boehm S/W Eng. paper	Project Management	
2	Thu 1/22/09	Pressman Ch. 4,5	Metrics, Project Planning	
3	Tue 1/27/09	Pressman Ch. 10	Systems Engineering	
3	Thu 1/29/09	Pressman Ch. 11, 12	Analysis Concepts, Analysis Modeling	

4	Tue 2/3/09	Pressman Ch. 12, 6	Analysis Modeling, Risk	
4	Thu 2/5/09	Pressman Ch. 6,8	Risk, SQA	**Ph.D. lecture topic due**
5	Tue 2/10/09	Pressman Ch. 7,9	Project Scheduling, SCM	Hand out project, start Phase I
5	Thu 2/12/09	Requirements Management in Software Processes: Rational Unified Process	Rational Unified Process: Best Practices for Software Development Teams (see URL below) The Ten Essentials of RUP	
6	Tue 2/17/09	Pressman Ch. 13, Parnas paper	Design Concepts	
6	Thu 2/19/09	Pressman Ch. 14, Wirth paper	Architecture Design	
7	Tue 2/24/09	Pressman Ch. 15,16	User Interface Design, Other Design Topics	
7	Thu 2/26/09	Pressman Ch. 19,20	Technical Metrics, OO Concepts	
8	Tue 3/3/09	None	**Project Presentations**	**Project Phase I due**, Start Phase II
8	Thu 3/5/09	None	**Project Presentations**	
9	Tue 3/10/09	No class		
9	Thu 3/12/09	Pressman Ch. 21,22	OOA, OOD	**Ph.D. lecture outline due**
10	3/16-3/21	No class – SPRING BREAK. Be careful, have fun		
11	Tue 3/24/09	Pressman Ch. 17, 18	Software Testing Techniques and Strategies, Exam review	
11	Thu 3/26/09	Pressman Ch. 18, 24, Chidamber paper	Software Testing Techniques and Strategies , OO Metrics	
12	Tue 3/31/09	Maintenance readings,	Software maintenance, Traceability	

		Brooks Ch. 16		
12	Thu 4/2/09	Pressman Ch. 25, Formal Specification: A Roadmap Axel van Lamsweerde, Kiczales paper	Formal Methods, formal specification techniques, AOP	
13	Tue 4/7/09	None	**Project Presentations**	**Project Phase II due,** Start Phase III
13	Thu 4/9/09	None	**Project Presentations**	
14	Tue 4/14/09	Pressman Ch. 27, 29, Frakes paper,	Component-Based Software Engineering, Web Engineering	
14	Thu 4/16/09	Experimental software eng. Readings, Brooks Ch. 17	Ethics, Experimental software engineering	
15	Tue 4/21/09	Assigned readings	**Ph.D. Student lectures**	
15	Thu 4/23/09	Software reliability reading	Software reliability, Review for Final	
16	Tue 4/28/09	None	**Project Presentations**	**Completed Project due**
16	Thu 4/30/09	None	**Project Presentations**	
Final	Thu 5/7/09 1300-1530	**Review all readings**	**Final**	**Final**

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Barry W. Boehm, Software Engineering, IEEE Trans. On Computers, 25(12):1226-1241, 19. http://ieeexplore.ieee.org/Xplore/defdeny.jsp?url=/iel5/12/35145/01674590.pdf?tp=&arnumber=1674590&isnumber=35145&code=2

Boehm, B. A Spiral Model for Software Development and Enhancement, Computer, Vol. 21, no. 5, May ’88, pp. 61-72. http://portal.acm.org/citation.cfm?id=12948

Parnas, D.L., On criteria to be used in decomposing systems into modules, CACM, vol. 15, no. 12, April '72, pp.1053-1058. http://portal.acm.org/citation.cfm?id=12948

Wirth, N. Program development by stepwise refinement, CACM, vol. 14, no. 4, 1971, pp. 221-227. http://www.acm.org/classics/dec95/

Musa, J.D., and Ackerman, A.F., Quantifying software validation: when to stop testing? IEEE SW, May 1989, pp. 19-27. http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=28120

Chidamber, S.R. and C.F. Kemerer, A metrics suite for object-oriented design, IEEE TSE, vol. SE-20, no. 6, June '94, pp.476-493. http://twiki.im.ufba.br/pub/Aside/ProjetoPibicCassio/CKMetrics.pdf

Frakes, W.B. and T.P. Pole, An empirical study of representation methods for reusable software components, IEEE TSE, vol SE-20, no. 8, Aug '94, pp. 617-630. http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=310671

Kiczales, G., Lamping, J. , Mendhekar, A., Maeda, C., Lopes, C.V., Loingtier, J.-M., and Irwin, J. Aspect--Oriented Programming. In European Conference on Object--Oriented Programming, ECOOP'97,
LNCS 1241, pages 220--242, Finland, June 1997. Springer--Verlag.
http://www2.parc.com/csl/groups/sda/publications/papers/Kiczales-ECOOP97/for-web.pdf

Rational Unified Process: Best Practices for Software Development Teams
http://www.augustana.ab.ca/~mohrj/courses/2000.winter/csc220/papers/rup_best_practices/rup_bestpractices.pdf

The Ten Essentials of RUP http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/dec00/TheTenEssentialsofRUPDec00.pdf

Formal Specification: A Roadmap
Axel van Lamsweerde http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalvanlamsweerde.pdf

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.

Syllabus for CS 499-001 Senior Design Project
Fall 2009

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours TR 0915 - 1000 (Robotics (CRMS) Bldg, Room 514D)

or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/CS499fall09.htm

Course:	CS 499 Senior Design Project
Section	001
Meets:	TR 0800 - 0915
Location:	Ralph G Anderson Ralph G Anderson-Rm.203-RGAN

Description:

This is a project course. Students will work in small groups to design and implement systems of current interest to computer scientists. The course will also provide a high-level overview of the software engineering discipline: software requirements, software design, software construction, software management, and software quality.

Course Outcomes:

- The student shall know the phases of the software development lifecycle and be able to define them. (C1)
- The student shall know the difference between project and process metrics. (C2)
- The student shall be able to define the terms version control and change control. (C3)
- The student shall be familiar with methods for performing requirements elicitation and requirements analysis. (C4)
- The student shall be able to discuss important design principles such as information hiding and abstraction. (C5)
- The student shall be able to discuss the differences between structured and object oriented analysis and design. (C6)
- The student shall be able to define key testing terms such as black box testing and white box testing. (C7)
- The student shall be able to perform the activities of the software lifecycle for a small to medium software project. (C8)
- The student shall be aware of ethical considerations in software engineering. (C9)

CS Outcomes:

Specifically, students will be able to:

CS1. use accepted project development processes in the project implementation

CS2. implement a large project

CS3. work as part of a team

CS4. present results of their work orally

CS5. document their work in a written report

Course Materials:

Required Text:
Shari Lawrence Pfleeger and Joanne M. Atlee
*Software Engineering: Theory and Practice,** **Fourth Edition***
Prentice Hall
ISBN:
You must obtain a copy of this text.

Recommended Texts:

Frederick P. Brooks, *Mythical Man Month, 2nd Edition*, Addison Wesley
ISBN: 0-201-83595-9

Martin Fowler and Kendall Scott
UML distilled: a brief guide to the standard object modeling language
(NOTE: 2nd edition available in library)

Gamma, Helm, Johnson & Vlissides
Design Patterns : elements of reusable object-oriented software
Addison-Wesley. ISBN 0-201-63361-2.

You do not have to obtain these, though you may choose to. Also, copies have been placed on reserve in the Engineering Library (3rd floor Anderson Hall)

Other readings, as assigned:

These are available via hyperlink in this syllabus or are on our course web page. See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 499 will be determined according to these weights:

Attendance and participation*:	10%
Presentations:	25%
Projects:	50%
Mid-term:	15%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

*Includes preparation of resume and completion of senior survey.

There will be a mid-term. The exam will be a 50 minute long "in class" exam. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work **between** groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. "All incidents of cheating and plagiarism are taken very seriously at this University. The minimum penalty for a first infraction is a zero on the assignment. [3]" **See attached policy on plagiarism, also here.**

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex. For information regarding these laboratories, see links under “facilities” from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group project for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. **Proper language usage is required.**

Schedule:

Week	**Date**	**Readings**	**Topics**	**Project, Exam**
1	Thu 8/27/09	Chapter 1	Why Software Engineering?, **Career Center Visitor**	
2	Tue 9/01/09	Chapter 2 (2.1-2), Boehm S/W Eng. paper	Process and Life Cycle	
2	Thu 9/03/09	Ch 11.1-11.3	Maintaining the System	
3	Tue 9/08/09	Ch 11.4-11.5	Maintaining the System	**Resumes due,** Hand out mini-project: Change Request
3	Thu 9/10/09	Ch 4	Requirements and Analysis	
4	Tue 9/15/09	Ch 4	Requirements and Analysis	
4	Thu 9/17/09	Ch 4	Requirements and Analysis	
5	Tue 9/22/09	Ch 5, Parnas Paper	Architecture (and HLD)	Change Request due.
5	Thu 9/24/09	Ch 6, Wirth Paper	Design	Hand out Phase I (Req/Anal/Arch)
6	Tue 9/29/09	Ch 6	Design	
6	Thu 10/01/09	Ch 6	Design	
7	Tue 10/06/09	**No class – work on projects**		
7	Thu 10/08/09	Ch 7	Writing the Program	
8	Tue 10/13/09	Ch 7	Writing the Program**, Midterm Review**	

8	Thu 10/15/09		**Midterm exam**	
9	Tue 10/20/09	Ch 8	Testing the Program	
9	Thu 10/22/09	Ch 8	Testing the Program	
10	Tue 10/27/09	None	**Project Presentations**	**Project Phase I due**, Start Phase II
10	Thu 10/29/09	None	**Project Presentations**	
11	Tue 11/3/09	Ch 8	Testing the Program	
11	Thu 11/5/09	Ch 9	Testing the System	
12	Tue 11/10/09	Ch 9	Testing the System	
12	Thu 11/12/09	**No class – work on projects**		
13	Tue 11/17/09	Ch 10	Delivering the System	
13	Thu 11/19/09	Ch 3	Planning and Managing the Project	
14	Tue 11/24/09	Ch 3	Planning and Managing the Project	
14	Thu 11/26/09	**No class – work on projects**		
15	Tue 12/1/09	Ch 12	Evaluating Products, Processes and Resources	**Senior surveys due**
15	Thu 12/3/09	Ch 12	Evaluating Products, Processes and Resources	
16	Tue 12/8/09	None	**Project Presentations**	**Phase II due**
16	Thu 12/10/09	None	**Project Presentations**	

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Barry W. Boehm, Software Engineering, IEEE Trans. On Computers, 25(12):1226-1241, 19. – see course web page

Boehm, B. A Spiral Model for Software Development and Enhancement, Computer, Vol. 21, no. 5, May ’88, pp. 61-72. - see course web page

Parnas, D.L., On criteria to be used in decomposing systems into modules, CACM, vol. 15, no. 12, April ’72, pp.1053-1058.   http://portal.acm.org/citation.cfm?id=361623

Wirth, N. Program development by stepwise refinement, CACM, vol. 14, no. 4, 1971, pp. 221-227. http://www.acm.org/classics/dec95/

Musa, J.D., and Ackerman, A.F., Quantifying software validation: when to stop testing? IEEE SW, May 1989, pp. 19-27. - see course web page

Chidamber, S.R. and C.F. Kemerer, A metrics suite for object-oriented design, IEEE TSE, vol. SE-20, no. 6, June ’94, pp.476-493.   http://portal.acm.org/citation.cfm?id=631131

Frakes, W.B. and T.P. Pole, An empirical study of representation methods for reusable software components, IEEE TSE, vol SE-20, no. 8, Aug ’94, pp. 617-630. - see course web page

Kiczales, G., Lamping, J. , Mendhekar, A., Maeda, C., Lopes, C.V., Loingtier, J.-M., and Irwin, J. Aspect--Oriented Programming. In European Conference on Object--Oriented Programming, ECOOP'97,
LNCS 1241, pages 220--242, Finland, June 1997. Springer--Verlag.
http://www2.parc.com/csl/groups/sda/publications/papers/Kiczales-ECOOP97/for-web.pdf

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.
[3] www.uky.edu/Ombud/acadoffenses/letterOfWarningExample.doc

Syllabus for CS 687 Empirical Software Engineering
Fall 2009

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours   TR 0915 - 1000 (Robotics (CRMS) Bldg, Room 514D)

or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/CS687-emp-sw-eng-fall09.htm
Course: CS 687 Empirical Software Engineering
Section: 002
Meets: TR 11:00 – 12:15
Location: Oliver H Raymond Building-Rm.C226-OHR

Description:

The course will present the following: Detailed study of the scientific process; particularly using the experimental method. Examination of how empirical studies are carried out in software engineering (by industry and by researchers). Review of the distinction between analytical techniques and empirical techniques. Study of when experimentation is required in software engineering, and what kinds of problems can be solved using experimentation. Examination of how to control variables and to eliminate bias in experimentation. Examination of analysis and presentation of empirical data for decision making. Students will learn how the scientific process should be applied, how and when to apply it in the software engineering area, and how to evaluate empirical evidence. The principles will be reinforced by examination of published experimental studies, and through designing and carrying out small experiments. On completion of the course, students will be in a position to design and carry out experiments in ways appropriate for a given problem, and will acquire skills in analyzing and presenting experimental data.

Course Outcomes:

Outcome 1 - The student shall know the scientific process
Outcome 2 - The student shall understand and be able to perform experimental design
Outcome 3 - The student shall understand the principles of experimental research and be able to carry out small experiments
Outcome 4 - The student shall be able to critically evaluate the empirical research carried out by others
Outcome 5 – The student shall be aware of ethical considerations in software engineering.

Course Materials:

Required Text:
Clases Wohlin, Per Runeson, Martin Host, Magnus C. Ohlsson, Bjorn Regnell, Anders Wesslen
Experimentation in Software Engineering: An Introduction
November 1999
Kluwer Academic Pub
ISBN: 0792386825
You must obtain a copy of Wohlin et al.

Other readings, as assigned: See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 687 will be determined according to these weights:

M.S. students:

Attendance and participation:	10%
Paper summaries:	20%
Team research project:	40%
Presentation:	30%

Ph.D. students:

Attendance and participation:	10%
Paper summaries:	20%
Individual research project:	40%
Presentation:	15%
Lecture Presentation:	15%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

Papers:

The first nine papers are about experimentation, and the rest are descriptions of experiments. It is important that you read the papers BEFORE the lectures, as the discussion will be very interactive. Turn in simple summaries and evaluations of **four** of the first nine **by Tuesday, 9/15/09**. For **one half** of the remaining papers (10), turn in a short (about one page) summary of the paper **by Tuesday 12/1/09**. The summaries should: (1) describe the problem in general terms, (2) paraphrase the experimental hypothesis, (3) summarize and critique the design, (4) discuss the conduct of the experiment, (5) explain whether the hypothesis was proved or disproved, and (6) critique the presentation of the paper. *Paper evaluations will be graded according to the following scale: 0: not submitted, 1: marginal, 2: what was expected, 3: outstanding. You are expected to have read all articles. Proper language usage is required.*

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and assignments with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points are good ways to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted **in person** at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work **between** groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. "All incidents of cheating and plagiarism are taken very seriously at this University. The minimum penalty for a first infraction is a zero on the assignment. [3]" **See attached policy on plagiarism, also here.**

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex, as well as the CSLab. For information regarding these laboratories, see links under “facilities” from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group projects for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor retains the right to make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. Proper language usage is required.

Schedule:

Week	Date	Readings	Topics	Project, Homework, Exam
1	Thu 8/27/09	Paper 1	Introduction, Overview of Scientific Method Lecture 1	

2	Tues 9/1/09	Paper 1,2	Lecture 1, Experimentation in Software Engineering, Lecture 2	
2	Thu 9/3/09	Papers 2 – 5, Wohlin Chapter 1, 2	Experimentation in Software Engineering, Lecture 2 – 4,	Hand out project assignment
3	Tues 9/8/09	Papers 6 - 9, Wohlin Chapter 3, 4, 5, 6, 7, 10	Experimentation in Software Engineering, Lecture 5 – 8, **Guest from Ag Center (Dr. Joe Chappell)**	
3	Thu 9/10/09	Papers 6 - 9, Wohlin Chapter 3, 4, 5, 6, 7, 10	Experimentation in Software Engineering, Lecture 5 - 8	
4	Tues 9/15/09	Papers 10, 11, 12, Wohlin Chapter 8, 9	Metrics and Complexity, **Guest from the Writing Center**	**Four summaries due,** Topic selection
4	Thu 9/17/09	Papers 10, 11, 12, Wohlin Chapter 8, 9	Metrics and Complexity, Ethics	
5	Tues 9/22/09	Wohlin Chapter 11, 12	Project Day	
5	Thu 9/24/09	Wohlin Chapter 11, 12	Project Day	
6	Tues 9/29/09	Papers 13, 14, 16	Testing, lecture Assert-Assess	Experiment Design Reviews
6	Thu 10/1/09	Papers 13, 14, 16	Testing, lecture Assert-Assess	Experiment Design Reviews
7	Tues 10/6/09	Papers 23a, 21, 22	Maintenance, lecture Writing	
7	Thu 10/8/09	Papers 23a, 21, 22	Maintenance, lecture Writing	
8	Tues 10/13/09	Papers 32, 33, 34, 35	Traceability	Hand out sample paper
8	Thurs 10/15/09	Papers 32, 33, 34, 35	Traceability	
9	Tues 10/20/09	Papers 23, 24	Requirements & Design	
9	Thurs 10/22/09	Papers 23, 24	Requirements & Design	Artifact Review
10	Tues 10/27/09	Papers 25, 26	Design	

10	Thurs 10/29/09	Papers 25, 26	Design	
11	Tues 11/3/09	Papers 27, 28	Design, Lecture Presentations	Draft paper due
11	Thurs 11/5/09	Papers 27, 28	Design, Lecture Presentations	
12	Tues 11/10/09	Papers 29, 30	HCI, Management and Inspections	Reviews due , Hand out sample presentation
12	Thu 11/12/09	Papers 29, 30	– No class – work on project	
13	Tues 11/17/09	None	HCI, Management and Inspections	
13	Thurs 11/19/09	None	Catch up	
14	Tues 11/24/09	None	Project Presentations	Final research papers due
14	Thurs 11/26/09	NO CLASS Have fun, be safe!		
15	Tues 12/1/09	None	Project Presentations	All reading paper summaries due
15	Thurs 12/3/09	None	Project Presentations	
16	Tues 12/8/09	None	Project Presentations	
16	Thurs 12/10/09	None	Project Presentations	
Final	Tues 12/15/09 1030 - 1300	None	Project Presentations – if time slot needed	

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Readings:

Empirical Methods Overview

35. **National Research Council**, *Academic Careers for Experimental Computer Scientists and Engineers, Ch. 1*, National Acadamy Press, pages 9-33, 1994. TOC PS
36. **Fenton**, Norman, Shari Lawrence Pfleeger and Robert L. Glass, "Science and Substance: A Challenge to Software Engineers", *IEEE Software*, V. 11, N. 4, pages 86-95, July 1994. Paper
37. **Tichy**, Walter F., "Hints for Reviewing Empirical Work in Software Engineering", *Empirical Software Engineering*, 5(4):309-312, December 2000. EMSE Home

38. **Amschler Andrews,** Anneliese and Arundeep S. Pradhan, "Ethical Issues in Empirical Software Engineering: The Limits of Policy", *Empirical Software Engineering*, 6(2):105-110, June 2001. EMSE Home
39. **Zendler**, Andreas, "A Preliminary Software Engineering Theory as Investigated by Published Experiments", *Empirical Software Engineering*, 6(2):161-180, June 2001. EMSE Home
40. **Harrison**, Warren "Editorial: Open Source and Empirical Software Engineering", *Empirical Software Engineering*, 6(3):193-194, September 2001. EMSE Home
41. **Shull**, Forrest, Manoel G. Mendoncça, Victor Basili, et al. "Knowledge-Sharing Issues in Experimental Software Engineering", *Empirical Software Engineering*, (9)1-2:111-137, March 2004. EMSE Home
42. **Karahasanovic',** Amela, Bente Anda, Erik Arisholm, Siw Elisabeth Hove, Magne Jørgensen, Dag I K Sjøberg and Ray Welland, "Collecting Feedback During Software Engineering Experiments", *Empirical Software Engineering*, 10(2):113-147, April 2005. EMSE Home
43. **Offutt**, Jeff, Yuan Yang and Jane Hayes, "SEEWeb: Making Experimental Artifacts Available", *Workshop on Empirical Research in Software Testing*, Boston, MA, July 2004. PDF

Metrics and Complexity

44. • **L. Briand** and J. Wust, "Empirical Studies of Quality Models in Object-Oriented Systems", *Advances in Computers*, vol. 56, 2002, Academic Press. Briand's homepage
45. • **Fenton**, Norman and Niclas Ohlsson, "Quantitative Analysis of Faults and Failures in a Complex Software System", *IEEE Transactions on Software Engineering*, (26)8:797-814, August 2000. PDF
46. • **Wohlin**, Claes, and Anneliese Amschler Andrews "Prioritizing and Assessing Software Project Success Factors and Project Characteristics using Subjective Data", *Empirical Software Engineering*, (8)3:285-308, September 2003. EMSE Home

Testing

47. • **Juristo**, Natalia, Ana M. Moreno, Sira Vegas "Reviewing 25 Years of Testing Technique Experiments", *Empirical Software Engineering*, (9)1-2:7-44, March 2004. EMSE Home
48. • **Ma**, Yu-Seung, Jeff Offutt and Yong Rae Kwon, "MuJava: An Automated Class Mutation System", *Journal of Software Testing, Verification and Reliability*, 15(2):97-133, June 2005. PDF
49. • **Roger T. Alexander** and Jeff Offutt, "Empirical Evaluation of Coupling-based Testing Techniques for Object-oriented Programs", submitted. PDF
50. • **Lionel C. Briand**, Massimiliano Di Penta and Yvan Labiche, "Assessing and Improving State-Based Class Testing: A Series of Experiments", *IEEE Transactions on Software Engineering*, 30(11), November 2004. PDF
51. • **Grindal**, Mats, Jeff Offutt and Jonas Mellin, "State-of-Practice: An Investigation of Testing Maturity", submitted. Preliminary version
52. • ~~**Stuart C. Reid**, "An Empirical Analysis of Equivalence Partitioning, Boundary Value Analysis and Random Testing", Proceedings of the 4th International Software Metrics Symposium (METRICS '97), 1997. PDF~~

Maintenance

53. • **Kajko-Mattsson**, Mira, "A Survey of Documentation Practice within Corrective Maintenance", *Empirical Software Engineering*, 10(1):31-55, January 2005. EMSE Home

54. • ~~**Liguo Yu**, Stephen R. Schach, Kai Chen and Jeff Offutt , "Categorization of Common Coupling and its Application to the Maintainability of the Linux Kernel", *IEEE Transactions on Software Engineering*, 30(10):694-706, October 2004. PDF local~~
55. • **Kai Chen**, Stephen R. Schach, Liguo Yu, Jeff Offutt and Gillian Z. Heller, "Open-Source Change Logs", *Kluwer's Empirical Software Engineering*, 9(3):197-210, September 2004. online EMSE Home
56. • **Stephen R. Schach**, Bo Jin, Liguo Yu, Gillian Z. Heller and Jeff Offutt, "Determining the Distribution of Maintenance Categories: Survey versus Measurement", *Kluwer's Empirical Software Engineering*, 8(4):351-365, December 2003. online EMSE Home

23a. Hassan, A. E. 2009. Predicting faults using the complexity of code changes. In Proceedings of the 2009 IEEE 31st international Conference on Software Engineering - Volume 00 (May 16 - 24, 2009). International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 78-88 PDF

Requirements

57. • **Damian**, Daniela, James Chisan, Lakshminarayanan Vaidyanathasamy and Yogendra Pal, "Requirements Engineering and Downstream Software Development: Findings from a Case Study", *Empirical Software Engineering*, (10)3:255-28, July 2005. EMSE Home

Design

58. • **Iris Reinhartz-Berger** and Dov Dori, "OPM vs. UML--Experimenting with Comprehension and Construction of Web Application Models", *Empirical Software Engineering*, 10(1), January 2005. EMSE Home
59. • **Marek Vokáccaron**, Walter Tichy, Dag I. K. SjØberg, Erik Arisholm and Magne Aldrin, "A Controlled Experiment Comparing the Maintainability of Programs Designed with and without Design Patterns-A Replication in a Real Programming Environment", *Empirical Software Engineering*, 9(3):149-195, September 2004. EMSE Home
60. • **Anda**, Bente and Dag I. K. Sjøberg, "Investigating the Role of Use Cases in the Construction of Class Diagrams", *Empirical Software Engineering*, (10)3:285-309, July 2005. EMSE Home
61. • **Svahnberg**, Mikael and Claes Wohlin "An Investigation of a Method for Identifying a Software Architecture Candidate with Respect to Quality Attributes", *Empirical Software Engineering*, (10)2:149-181, April 2005. EMSE Home
62. • **Knight**, John C. and Nancy G. Leveson, "An Experimental Evaluation of the Assumption of Independence in Multiversion Programming", *IEEE Transactions on Software Engineering*, (SE-12)1:96-109, January 1986. NEC Research Index (CiteSeer)

HCI

63. • **Miara**, Richard J., Joyce A. Musselman, Juan A. Navarro, and Ben Shneiderman, "Program Indentation and Comprehensibility", *Communications of the ACM*, (26)11:861-867, November 1983. ACM

Management and Inspections

64. • **McDonald**, James, "The Impact of Project Planning Team Experience on Software Project Cost Estimates", *Empirical Software Engineering*, (10)2:219-234, April 2005. EMSE Home

65. • ~~**Thelin**, Thomas, Per Runeson, Claes Wohlin, et al. "Evaluation of Usage-Based Reading-Conclusions after Three Experiments", *Empirical Software Engineering*, (9)1-2:77-110, March 2004. EMSE Home~~

Traceability

66. O.C.Z. Gotel and A.C.W. Finkelstein. An analysis of the requirements traceability problem. In 1st International Conference on Requirements Engineering, pages 94--101, 1994. PDF
67. Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., and Merlo, E. Recovering Traceability Links between Code and Documentation. IEEE Transactions on Software Engineering, Volume 28, No. 10, October 2002, 970-983. PDF
68. Jane Huffman Hayes, Alex Dekhtyar: A Framework for Comparing Requirements Tracing Experiments. International Journal of Software Engineering and Knowledge Engineering 15(5): 751-782 (2005) PDF
69. Jane Cleland-Huang, Raffaella Settimi, Oussama Ben Khadra, Eugenia Berezhanskaya, Selvia Christina: Goal-centric traceability for managing non-functional requirements. ICSE 2005: 362-371 PDF

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.
[3] www.uky.edu/Ombud/acadoffenses/letterOfWarningExample.doc

Syllabus for CS 499-001 Senior Design Project
Fall 2010

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building
Office hours TR 0915 - 1000 (Robotics (CRMS) Bldg, Room 514D) or by appointment

Course information:

Course homepage http://selab.netlab.uky.edu/homepage/CS499fall10.htm

Course:	CS 499 Senior Design Project
Section	001
Meets:	TR 0800 - 0915
Location:	Ralph G Anderson Ralph G Anderson-Rm.203-RGAN

Description:

This is a project course. Students will work in small groups to design and implement systems of current interest to computer scientists. The course will also provide a high-level overview of the software engineering discipline: software requirements, software design, software construction, software management, and software quality.

Course Outcomes:

- The student shall know the phases of the software development lifecycle and be able to define them. (C1)
- The student shall know the difference between project and process metrics. (C2)
- The student shall be able to define the terms version control and change control. (C3)
- The student shall be familiar with methods for performing requirements elicitation and requirements analysis. (C4)
- The student shall be able to discuss important design principles such as information hiding and abstraction. (C5)
- The student shall be able to discuss the differences between structured and object oriented analysis and design. (C6)
- The student shall be able to define key testing terms such as black box/white box testing. (C7)
- The student shall be able to perform the activities of the software lifecycle for a small to medium software project. (C8)
- The student shall be aware of ethical considerations in software engineering. (C9)

<u>CS Outcomes:</u>

Specifically, students will be able to:
CS1. use accepted project development processes in the project implementation
CS2. implement a large project
CS3. work as part of a team
CS4. present results of their work orally
CS5. document their work in a written report

Course Materials: **Required Text:**

Shari Lawrence Pfleeger and Joanne M. Atlee
*Software Engineering: Theory and Practice,*** Fourth Edition****
Prentice Hall
ISBN: 0136061699
You must obtain a copy of this text.

Recommended Texts:

Frederick P. Brooks, *Mythical Man Month, 2nd Edition*, Addison Wesley
ISBN: 0-201-83595-9

Martin Fowler and Kendall Scott
UML distilled: a brief guide to the standard object modeling language
(NOTE: 2nd edition available in library)

Gamma, Helm, Johnson & Vlissides
Design Patterns : elements of reusable object-oriented software
Addison-Wesley. ISBN 0-201-63361-2.

You do not have to obtain these, though you may choose to. Also, copies have been placed on reserve in the Engineering Library (3rd floor Anderson Hall)

Other readings, as assigned:

These are available via hyperlink in this syllabus or are on our course web page. See list below.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 499 will be determined according to these weights:

Attendance and participation*:	10%
Presentations:	15%
Projects:	60%
Mid-term:	15%

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

*Includes preparation of resume and completion of senior survey.

There will be a mid-term. The exam will be an "in class" exam. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Attendance:

Students are expected to attend and participate in all scheduled classes. Arrival after attendance has been taken at the start of class will be considered an absence. The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class starts are **late**. Credit will be deducted for late assignments. Assignments will not be accepted after solutions have been distributed.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work **between** groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. "All incidents of cheating and plagiarism are taken very seriously at this University. The minimum penalty for a first infraction is a zero on the assignment. [3]" **See attached policy on plagiarism, also here.**

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

> Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department and located in Room 203 of the Engineering Annex. For information regarding these laboratories, see links under "facilities" from the Computer Science homepage (http://www.cs.uky.edu/). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Group Projects:

The group project for the course will require you to work together with other students in the class. You will be evaluated on your contribution to the group project and presentations of the project results. The instructor will make group assignments. Group members are not guaranteed to receive the same grade; evaluation of the group will be individualized to determine individual understanding, commitment, and mastery of the project goals. As part of the project, written reports will be required. **Proper language usage is required.**

Schedule:

Week	**Date**	**Readings**	**Topics**	**Project, Exam**
1	Thu 8/26/10	Chapter 1	Why Software Engineering?, **Career Center Visitor**	
2	Tue 8/31/10	Chapter 2 (2.1-2), Boehm S/W Eng. paper	Process and Life Cycle	
2	Thu 9/02/10	Ch 3	Planning and Managing the Project	
3	Tue 9/07/10	Ch 3	Planning and Managing the Project	**Resumes due**
3	Thu 9/09/10	Ch 4	Requirements and Analysis	
4	Tue 9/14/10	Ch 4	Requirements and Analysis	
4	Thu 9/16/10	Ch 5, Parnas Paper	Architecture (and HLD)	

5	Tue 9/21/10	Ch 6, Wirth Paper	Design	Hand out Phase I assignment
5	Thu 9/23/10	Ch 6	Design	
6	Tue 9/28/10	Ch 6	Design	
6	Thu 9/30/10	**No class – work on projects**		
7	Tue 10/05/10	Ch 7	Writing the Program	
7	Thu 10/07/10	Ch 7, 8	Writing the Program, Testing the Program	
8	Tue 10/12/10	None	**Project Presentations**	**Project Phase I due**, Start Phase II
8	Thu 10/14/10	Ch 8	Testing the Program	
9	Tue 10/19/10	Ch 8	Testing the Program**, Midterm Review**	
9	Thu 10/21/10		**Midterm exam**	
10	Tue 10/26/10	Ch 8, 9	Testing the Program, Testing the System	
10	Thu 10/28/10	Ch 9	Testing the System	
11	Tue 11/2/10	Ch 10	Delivering the System	
11	Thu 11/4/10	Ch 11.1-11.3	Maintaining the System	
12	Tue 11/09/10	**No class – work on projects**		
12	Thu 11/11/10	Ch 11.4-11.5	Maintaining the System	
13	Tue 11/16/10	None	**Project Presentations**	**Project Phase II due,** Start Phase III

13	Thu 11/18/10	Ch 12, Code of Conduct	Evaluating Products, Processes and Resources, Ethics	
14	Tue 11/23/10	**No class – work on projects**		
14	Thu 11/25/10	***No class***	***Fall break - Thanksgiving***	
15	Tue 11/30/10	Ch 12	Evaluating Products, Processes and Resources	**Senior surveys due**
15	Thu 12/2/10	Ch 14	Future of Software Engineering	
16	Tue 12/7/10	None	**Project Presentations**	**Completed projects due**
16	Thu 12/09/10	None	**Project Presentations**	

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

Possible outside readings:

Barry W. Boehm, Software Engineering, IEEE Trans. On Computers, 25(12):1226-1241, 19. – see course web page

Boehm, B. A Spiral Model for Software Development and Enhancement, Computer, Vol. 21, no. 5, May ’88, pp. 61-72. - see course web page

Parnas, D.L., On criteria to be used in decomposing systems into modules, CACM, vol. 15, no. 12, April ’72, pp.1053-1058. http://www.cs.umd.edu/class/spring2003/cmsc838p/Design/criteria.pdf

Wirth, N. Program development by stepwise refinement, CACM, vol. 14, no. 4, 1971, pp. 221-227. http://sunnyday.mit.edu/16.355/wirth-refinement.html

Musa, J.D., and Ackerman, A.F., Quantifying software validation: when to stop testing? IEEE SW, May 1989, pp. 19-27. - see course web page

Chidamber, S.R. and C.F. Kemerer, A metrics suite for object-oriented design, IEEE TSE, vol. SE-20, no. 6, June ’94, pp.476-493. http://portal.acm.org/citation.cfm?id=631131

Frakes, W.B. and T.P. Pole, An empirical study of representation methods for reusable software components, IEEE TSE, vol SE-20, no. 8, Aug ’94, pp. 617-630. - see course web page

Kiczales, G., Lamping, J. , Mendhekar, A., Maeda, C., Lopes, C.V., Loingtier, J.-M., and Irwin, J. Aspect--Oriented Programming. In European Conference on Object--Oriented Programming, ECOOP'97,
LNCS 1241, pages 220--242, Finland, June 1997. Springer--Verlag.
http://www2.parc.com/csl/groups/sda/publications/papers/Kiczales-ECOOP97/for-web.pdf

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.
[3] www.uky.edu/Ombud/acadoffenses/letterOfWarningExample.doc

Syllabus for CS 585-001 Software Testing

Instructor:

Dr. Jane Hayes (www.cs.uky.edu/~hayes).
Room 233**,** Hardymon Building 257-3171
Office hours 1:00 – 1:50 pm T, 9:30-10:30 R or by appointment in Robotics (CRMS) Room 514D

Course information:

Course homepage
http://selab.netlab.uky.edu/homepage/CS585_softwaretest_spr11.htm

Course:	CS 585 Software Testing
Section:	001
Meets:	TR 2:00 – 3:15
Location:	**Gatton Business and Economics Bldg room 306 – NEW ROOM!**

Description:

Concepts and techniques for testing software and assuring its quality. Topics cover software testing at the unit, module, subsystem, and system levels, automatic and manual techniques for generating and validating test data, the testing process, static vs. dynamic analysis, functional testing, inspections, and reliability assessment. *Professor's note*: The course will attempt to prepare students to test software in structured, organized ways. This course should provide practical knowledge of a variety of ways to test software, an understanding of some of the tradeoffs between testing techniques, and a feel for the practice of software testing **and** the research in software testing.

Course outcomes:

The student shall know the levels of testing and be able to define them. (C1)
The student shall understand the test process, including how to develop test plans, test cases, and test reports (results). (C2)
The student shall know the definitions of key testing terms such as coverage criterion, subsumption, and black box testing. (C3)
The student shall understand the difference between a fault and a failure. (C4)

The student shall understand how to generate test cases that achieve data flow coverage and/or determine if given test cases achieve data flow coverage for a given program. (C5)

The student shall know how to apply at least two test techniques, e.g. equivalence partitioning testing, random testing, control-flow based criteria testing, data-flow based criteria testing, mutation testing, object-oriented testing. (C6)

The student shall know how to evaluate and/or report test related measures such as coverage/thoroughness measures; remaining number of defects/Fault density; mutation score; reliability growth; etc. (C7)

The student shall understand class integration test order. (C8)

The student shall be aware of ethical considerations in software testing. (C9)

Course Materials:

Required Texts:

- Paul Ammann and Jeff Offutt, *Introduction to Software Testing*, Cambridge University Press, Cambridge, UK, ISBN 0-52188-038-1, 2008.
- CAST_CBOK_Ver-6-2-1 Certified Associate Software Tester (CAST) Common Body of Knowledge (CBOK), Version 6.2 © 2008, Quality Assurance Institute. – details will be given in class on downloading this document
- Various artifacts available over the web. Details as needed.

Other readings, as assigned: See end of syllabus, there may be additions.

Course web page:

Course materials will be available on the course web page. The course web page and e-mail will be important methods of distributing information for the course.

Grading:

Your grade in CS 585 will be determined according to these weights:

<u>Undergraduate students:</u>

Homework assignments/paper summaries:		26%
Quizzes:		22%
Project:		42%
Final:	10%	

<u>Graduate students:</u>

Homework assignments/paper summaries:		19%
Additional paper summaries:	5%	
Quizzes:		19%
Project:		35%
Short paper:	12%	
Final:	10%	

Where:

A=	92 - 100%
B =	83 - 91%
C=	74 - 82%
D=	65 - 73%
F =	64 and below

The lowest homework grade will be dropped; allowing for a missed class.

There will be one closed book, in-class, comprehensive final. The date for the exam is listed in the schedule below.

Whining Lowers Grades [1]:

You are always welcome and encouraged to discuss exams and homework with your professor; it is an excellent way to learn from your mistakes. If the grading does not make sense to you, please ask. You may not yet have understood your mistake -- or there may be an error in the grading. However, whining, demanding a re-grade instead of requesting one, or saying that you deserve more points is a good way to convince a professor to re-grade your entire assignment or exam, perhaps with more careful attention to your mistakes.

Late Policy:

Assignments must be submitted in person at or before **class time** on the day the assignment is due. Assignments turned in after class has started are **late and will have points deducted. Assignments will not be accepted after solutions have been distributed**.

Attendance/Absences:

Arrival after class has started is disruptive to the class and is not appreciated. Arrival after student presentations have begun will not be allowed. **Absence the day of a quiz, test or presentation constitutes a 0 for that grade element.** The following are acceptable reasons for excused absences: 1) serious illness; 2) illness or death of family member; 3) University-related trips (S.R. 5.2.4.2.C); 4) major religious holidays; 5) other circumstances that the instructor finds to be "reasonable cause for nonattendance." It is the student's responsibility to provide sufficient documentation regarding the nature of the absence, and the instructor retains the right to ask for such proof.

Academic Honor Code:

Individual work (homework, exams) must be your own. No sharing of computer code or other work will be allowed. Group projects allow the sharing of ideas and computer code within the group. No sharing of work between groups will be acceptable. The University of Kentucky's guidelines regarding academic dishonesty will be strictly enforced. **Note that the minimum penalty for plagiarism is an E in the course.** A link to a paper "Plagiarism: What is it?" may be found at the UK Ombud web site or can be accessed at http://www.uky.edu/Ombud/Plagiarism.pdf.

Accepting Responsibility for Failure [2]:

Failure is an unpleasant word, with bleak connotations. Yet it is a word that applies to every one of us at different stages of our lives. No one is exempt. Our icons, gurus, religious leaders, politicians, rock stars and educators all fail. It is simply a reality of being human. It is also a label that we fight desperately to avoid. And it is this fight to avoid failure that drives us forward towards our life accomplishments. So--why can't we take responsibility for our own failure when it does occur?

We need to accept responsibility for a very important reason--namely, maturity. We cannot reach a full level of maturity until we accept ownership of our own mistakes. As an educator, I am confronted with this problem on a daily basis. When a student is late for class, it is because a parent failed to wake them up. A failed test becomes the responsibility of the teacher, the system, society, an after school job, but never the fault of the test taker. An incomplete assignment is inevitably due to the needy demands of a friend, or an electrical failure. I feel particularly blessed because the power circuits leading to my home must be exceptionally fine, as I have yet to experience the myriad of blackouts that have plagued my students.

Nevertheless, the daily onslaught of excuses has left me questioning the value of our education system. What, after all, is the point of "higher learning" if we fail to master the basic task of owning up to our own mistakes?

As we proceed through our education system and indeed life, our excuses for failure become more grandiose and perhaps more grotesque because the crude reality is that we have failed to mature in any significant sense of the word. To continually shift responsibility away from ourselves is worse than being a coward. Even a coward will admit that their failure is a result of their own lack of courage.

Accepting failure takes strength of character, honesty and humility. It provides a building block for future achievements. When we deny culpability, we rob ourselves of the chance to learn from our mistakes. We condemn ourselves to a lifetime pattern of avoidance and deception. Like Marley's ghost, dragging his chains of missed humanitarian opportunities behind him, we crawl forward pulling our chains of pathetic excuses behind us--never fully maturing, never fully reaching our true potential. This stale baggage is far more character eroding than any of our individual failures could ever be.

Computer Facilities:

You will be assigned an account for this course in the Multilab, a PC laboratory administered by the Computer Science department, located in Room 203 of the Engineering Annex as well as the CSLab. For information regarding these laboratories, see links under “facilities” from the Computer Science homepage (www.cs.uky.edu). You may use alternative computer systems for developing and testing your work, provided that your submitted work will compile and run under the proper software environment as directed in class.

Paper Summaries [3]:

You are required to read and evaluate each of the assigned readings prior to discussion in class. Summaries/evaluations are **due at the start of class**. **Late paper summaries will have points deducted. Paper summaries will not be accepted after graded papers have been returned.** Each paper summary should be on a separate sheet of paper (**ONE page** only), and include:

· The title and first author's name
· The main point that the article seemed to make (2-5 sentences)
· Two subjective numerical ratings on a 1-to-6 scale (1 low, 6 high):
a) How important is the material covered in the article?
b) How well-written was the article?
· Two to three paragraphs concerning the content of the article, containing either:

a) A question about the article, such as one that you or someone reading the paper for the first time might have to stop and study, look elsewhere, or re-read to find an answer. Questions should be accompanied by an elaboration of the question, and/or a discussion of its relevance.

b) A comment on the article, such as discussion of application, classification, comparison, and/or evaluation of method or methods.

c) What you liked, disliked, found interesting or found unclear in the article.

Also, see "How to Read an Engineering Research Paper" by Bill Griswold for additional ideas (http://www-cse.ucsd.edu/users/wgg/CSE210/howtoread.html).

Paper evaluations will be graded according to the following scale: 0: not submitted, 1: marginal, 2: what was expected, 3: outstanding. You are expected to have read all articles. Proper language usage is required.

Homework assignments:

There will be small computer and homework assignments for some of our topics. Some will require pencil and paper and others will require modest programming or use of tools available from the Web. Proper language usage is required.

Quizzes:

We will have periodic quizzes and no midterm exam. Quizzes will be given during the **first 10 - 20 minutes** of each class and no makeup or late quizzes will be given.

Short Paper and/or Presentation:

Graduate students will prepare a 3 to 4 page paper that describes a current testing topic of the student's choosing. Proper language usage is required.

Testing Project:

Students will develop and execute a testing approach for a software application of their choosing (there are many open source applications available, the professor also has access to three games: Mole Whacker, Spell Dodger, and Set). Students will develop a Test Plan, Test Description (test cases), and Test Report. Status reports may be due at various points to document progress. More details will be provided in class.

Schedule:

Week	Date	Readings	Topics	Assignments, Exam
1	Thu 1/13/11	Ammann/Offutt Chapter 1.1-	Introduction and Background on Coverage	

		1.3, CBOK 1.1 – 1.7	Based Testing	
2	Tue 1/18/11	Ammann/Offutt Chapter 1.1-1.3, (Read Ariane 5 Flight Failure– graduate students), CBOK 1.1 – 1.7	Introduction and Background	Graduate student paper summary due
2	Thu 1/20/11	22 CBOK 2, 3	23 Test Environment, Test Management – visitor?	Homework 1 (1st) due
3	Tue 1/25/11	24 CBOK 2, 3	25 Test Environment, Test Management	Quiz 1
3	Thu 1/27/11	CBOK 4	Test Planning	Will hand out graduate paper assignment
4	Tue 2/1/11	AO 2.1-2.2, (Read Offutt paper – graduate students)	Graph Coverage	Graduate student paper summary due
4	Thu 2/3/11	AO 2.1-2.2	Graph Coverage	Homework 2 (2nd) due
5	Tue 2/8/11	CBOK 5, 6	Test Execution, Test Reporting	Will hand out testing project assignment
5	Thu 2/10/11	AO 2.3-2.5	Graph Coverage for code – to RGAN 102, start Homework 4 in lab	Topics for graduate papers due
6	Tue 2/15/11	AO 2.3-2.5	Graph Coverage for other artifacts – visitor?	Quiz 2
6	Thu 2/17/11	Web readings	Game testing– to RGAN 102	
7	Tue 2/22/11	AO 3.1, 3.2, 3.3, 3.5	Logic testing	Project proposal due
7	Thu 2/24/11	AO 3.1, 3.2, 3.3, 3.5	Logic testing	Homework 6 (3rd) due
8	Tue 3/1/11	AO 4**,** Read Hutchins et al.	Input space partitioning	Paper Summary due

8	Thu 3/3/11	AO 4, Killer Robot article	Input space partitioning, Ethics	26 Homework 8 (4th) due
9	Tue 3/8/11	AO 5.1-5.3, 5.5	Syntax testing/mutation analysis, Integration and OO, input space grammars	MIDTERM OF SEMESTER
9	Thu 3/10/11	AO 5.1-5.3, 5.5	Syntax testing/mutation analysis, Integration and OO, input space grammars – to RGAN 102	Test Plan/Description due
10	Tue 3/15/11	NO CLASS	NO CLASS	NO CLASS
10	Thu 3/17/11	NO CLASS	NO CLASS	NO CLASS
11	Tue 3/22/11	27 AO 6.1, 6.2	28 Practical considerations technologies – visitor?	Homework 9 (5th) due
11	Thu 3/24/11	AO 6.3, 8.1	Practical considerations technologies, Building tools	Homework 4 (6th) due
12	Tue 3/29/11	AO 6.3, 8.1	Practical considerations technologies, Building tools	Draft graduate papers due
12	Thu 3/31/11	AO 7.1, 7.2	Testing OO software– to RGAN 102	Quiz 3
13	Tue 4/5/11	AO 7.1, 7.2, CBOK 1.8, web readings	Testing OO software, coding standards, split testing, fuzz testing	Project Status report due (preliminary test report)
13	Thu 4/7/11	29 CBOK 6, 7, 8, 9, buffer overflow reading	30 Test Reporting, User Acceptance testing, Testing Contractor Software, Testing Software Controls, buffer overflow, information flow analysis	
14	Tue 4/12/11	31 CBOK 7, 8,	32 User Acceptance testing, Testing	Quiz 4

		9	Contractor Software, Testing Software Controls	
14	Thu 4/14/11	33 CBOK 10, CBOK Vocabulary	34 Testing New Technologies, Vocabulary – to RGAN 102	Graduate papers due
15	Tue 4/19/11	35 CBOK 10, CBOK Vocabulary	36 Testing New Technologies, Vocabulary – visitor?	
15	Thu 4/21/11	37 Web readings	38 Test driven development (agile testing)	Project due – Test Reports
16	Tue 4/26/11	39 Read Frankl and Weiss, web readings	40 Web testing	Paper Summary due
16	Thu 4/28/11		Course Wrap up, Final Exam Review	
Final	Fri 5/6/2011 0800-1030	**Review all readings, assignments**		**Exam (Final)**

The syllabus is subject to change, and you are responsible for keeping informed of any alterations.

<u>Possible outside readings:</u>

Report by the Inquiry Board. ARIANE 5 Flight 501 Failure. Technical report, European Space Agency, 1996. J.L. Lions, Chairman of the Board.
http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html

Experiments of the effectiveness of dataflow- and controlflow-based test adequacy criteria
International Conference on Software Engineering
Proceedings of the 16th international conference on Software engineering
Sorrento, Italy, Pages: 191 – 200,   Year of Publication: 1994, ISBN:0-8186-5855-X
Authors**: Monica Hutchins,** Herb Foster, Tarak Goradia, Thomas Ostrand
Paper available from our course web page

An experimental comparison of the effectiveness of the all-uses and all-edges adequacy criteria   International Symposium on Software Testing and Analysis
Proceedings of the symposium on Testing, analysis, and verification, Victoria, British Columbia, Canada
Pages: 154 – 164, Year of Publication: 1991, ISBN:0-89791-449-X
Authors: Phyllis G. **Frankl,** Stewart N. **Weiss**
Paper available from our course web page

41   Investigations of the software testing coupling effect
ACM Transactions on Software Engineering and Methodology (TOSEM)
Volume 1 , Issue 1 (January 1992), Pages: 5 – 20, Year of Publication: 1992
ISSN:1049-331X
Author: A. Jefferson Offutt
Paper available from our course web page

[1] Dr. Judy Goldsmith
[2] http://www.scs.ryerson.ca/~dwoit/failure.html.
[3] http://www.cs.ucsc.edu/~ejw/courses/290gw02/assignments.htm