

# CSci 690: Reading Course – Software Engineering Education

## Where and when

- Wed 2-4. McGloughlin-Street Hall 120.

## Instructor

- David Coppit (<http://www.cs.wm.edu/~coppit/>, [coppit@cs.wm.edu](mailto:coppit@cs.wm.edu))

## Website

- <http://www.cs.wm.edu/~coppit/csci690/>

## Description

- This reading course introduces students to current practices in software engineering education. Topics include: software engineering curricula, software engineering's relationship to computer science and other disciplines, "professionalization" of software engineering, body-of-knowledge efforts, and teaching software engineering.
- Students will be expected to participate in weekly discussions on the selected papers. Before each class, students email the instructor a paragraph for each paper they have read.
- With the instructor, each student will also select a special topic to serve as the basis for a term paper due at the end of the course. Intermediate milestones (e.g. abstract, drafts) will be set during the course.

## Students

- Jennifer Haddox-Schatz
- Ashwin Mundra

## Grading

Pass/Fail

## Schedule

1/21	Introduction and planning
1/27	Introduction <ul style="list-style-type: none"><li>• Mary Shaw, Software Engineering Education: A Roadmap. 9 pages. se-ed3.pdf</li><li>• Meyer, Bertrand. 2001. Software Engineering in the Academy. IEEE Computer. (May), 28-35. 8 pages. r5028.pdf</li></ul>

2/4	<p>Professionalization</p> <ul style="list-style-type: none"> <li>• Maibaum, What We Teach Software Engineers in the University: Do We Take Engineering Seriously? Deals with the "is SE really engineering?" issue. 6 pages. p40-maibaum.pdf</li> <li>• ACM, A Summary of the ACM Position on Software Engineering as a Licensed Engineering Profession. 8 pages. selep_main.pdf</li> <li>• Meng Seet, Certification for Computing Graduates. Is it a good idea? 2 pages. seet00certification.ps</li> </ul>
2/11	<p>Software Engineering's Relationship with CS and Other Engineering Disciplines</p> <ul style="list-style-type: none"> <li>• Parnas, D., "Software Engineering Programmes are not Computer Science Programs", draft paper, Department of Computing and Software, McMaster University, 1998. 16 pages. parnas98software.ps</li> <li>• Nancy G. Leveson, High-Pressure Steam Engines And Computer Software. 13 pages. steam.pdf</li> <li>• David Lorge Parnas, Software Engineering: An Unconsummated Marriage. 1 page. marriage.pdf</li> <li>• Denning, Peter. 1998. Computer Science and Software Engineering: Filing for Divorce? ACM Communications 41 (July). 1 page. p128-denning.pdf</li> </ul>
2/18	<p>Body of Knowledge Efforts: Part 1</p> <ul style="list-style-type: none"> <li>• Fairley and Tripp, "Software Engineering: Moving From Craft to Profession" 8 pages. CraftToProfession.pdf</li> <li>• J. Barrie Thompson, Colin J. Hardy. Use and Evaluation of SWEBOK by Postgraduate Students. Conference on SE Education/Training 2002. 10 pages. 15150066.pdf</li> </ul>
2/25	<p>Body of Knowledge Efforts: Part 2</p> <ul style="list-style-type: none"> <li>• SWEBOK, Guide to the Software Engineering Body of Knowledge. <a href="http://www.swebok.org">http://www.swebok.org</a>. 100 pages. SWEBOK.pdf</li> <li>• David Notkin, Michael Gorlick, and Mary Shaw, An Assessment of Software Engineering Body of Knowledge Efforts. 8 pages. bok_assessment.pdf</li> </ul>

3/3	<p>Software Engineering Curricula</p> <ul style="list-style-type: none"> <li>• J. Cowling What Should Graduating Software Engineers Be Able To Do? Conference on SE Education/training 2003. 8 pages. 18690088.pdf</li> <li>• Michael J. Lutz, J. Fernando Naveda, The road less traveled: a baccalaureate degree in software engineering. 5 pages. Short. p287-lutz.pdf</li> <li>• CCSE, Computing Curricula Software Engineering. This is a big website - we'd have to determine which portions to look at. <a href="http://sites.computer.org/ccse/">http://sites.computer.org/ccse/</a>. 100 pages. ccse.pdf</li> </ul>
3/10	<b><i>Spring Break</i></b>
3/17	<p>Teaching Software Engineering: Part 1</p> <ul style="list-style-type: none"> <li>• Mary Shaw, We Can Teach Software Better. 5 pages. crn.pdf</li> <li>• A Debate on Teaching Computer Science. P. Denning, ed. A debate on the subject including Parnas and Dijkstra as participants. 15 pages. p1397-denning.pdf</li> <li>• Margot Postema, Jan Miller, Martin Dick, Including Practical Software Evolution in Software Engineering Education. Conference on SE Education/Training (2001). Discusses teaching the notion of software maintenance to undergrads. 7 pages. 10590127.pdf</li> </ul>
3/24	<p>Teaching Software Engineering: Part 2</p> <ul style="list-style-type: none"> <li>• Michael Gnatz, Leonid Kof, Franz Prilmeier, Tilman Seifert A Practical Approach of Teaching Software Engineering. Conference on SE Education/training 2003. Addresses "social" issues encountered in SE - e.g. dealing with clients, other team members, etc. 7 pages. 18690120.pdf</li> <li>• W. R. Adrion. Developing and Deploying Software Engineering Courseware in an Adaptable Curriculum Framework. In Proc. Of the International Conference on Software Engineering (2000). 10 pages. adaptable.pdf</li> </ul>

3/31	<p>Special Topic: Projects in Software Engineering Classes: Part 1</p> <ul style="list-style-type: none"> <li>• Mats Daniels, et. al. Open Ended Group Projects, Motivating Students and Preparing them for the "Real World". Conference on SE Education/training (2002). 10 pages. 15150128.pdf</li> <li>• Christine Mingins, Jan Miller, Martin Dick and Margot Postema. How we teach software engineering. JOOP. February 1999 Vol. 11, No. 9, 64-75, 1999. Describes an SE course taught to undergrads - course involves a large project and discusses the project - seems interesting! 5 pages. mingins.pdf</li> <li>• Jeff Zhang, Dolores Zage, Wayne Zage. Improving Project Planning/Tracking for Student Software Engineering Projects through SOPPTS. Conference on SE Education/training 2003. Describes on-line project planning/tracking system for student SE projects. 7 pages. 18690185.pdf</li> </ul>
4/7	<p>Special Topic: Projects in Software Engineering Classes: Part 2</p> <ul style="list-style-type: none"> <li>• Mary Shaw and Jim Tomayko, Models for Undergraduate Project Courses in Software Engineering. 30 pages. tr10.91.pdf</li> </ul>
4/14	<p>Special Topic: Teaching Software Design: Part 1</p> <ul style="list-style-type: none"> <li>• David Carrington, Teaching Software Design and Testing. 4 pages 1049.pdf</li> <li>• David Carrington, Teaching Software Design with Open Source Software. 6 pages 1273.pdf</li> <li>• Sims-Knight and Upchurch, "Teaching Object-oriented Design Without Programming: A Progress Report", 13 pages. compscied.pdf</li> </ul>
4/21	<p>Special Topic: Teaching Software Design: Part 2</p> <ul style="list-style-type: none"> <li>• Kaufman, Felder, and Fuller, "Accounting for Individual Effort in Cooperative Learning Teams." 10 pages Kaufmanpap.pdf</li> <li>• Huang and Swenton, "Teaching undergraduate software design in a liberal arts environment using RoboCup", 5 pages p114-huang.pdf</li> <li>• Bishop-Clark and Kipper, "An Undergraduate Course in Object-Oriented Software Design" 5 pages. 1356.pdf</li> </ul>
4/28	<p>Maturation of Software Engineering as a Discipline</p> <ul style="list-style-type: none"> <li>• Anthony I. Wasserman, Toward a Discipline of Software Engineering. 8 pages. s6023.pdf</li> <li>• Mary Shaw, Prospects for an Engineering Discipline of Software. 10 pages. s6015.pdf</li> </ul>
5/10	<p>Final Paper Due</p>

