PURDUE UNIVERSITY | Elmore Family School of Electrical and Computer Engineering

Engineering Faculty Document No. EFD 46-22
February 1, 2022

Memorandum

To: The College of Engineering Faculty

From: The Elmore Family School of Electrical and Computer Engineering

Re: Course modifications to ECE 46100 Software Engineering

The faculty of the Elmore Family School of Electrical and Computer Engineering has approved the changes to the following undergraduate course. This action is now submitted to the Engineering Faculty with a recommendation for approval.

FROM

ECE 46100 Software Engineering, Sem. 1, Class 3, Lab 0, Cr. 3.
Prerequisites: ECE 30862

Introduction to software engineering principles, with special emphasis on the process, methods, and tools needed to develop and test quality software products and systems.

No learning outcomes listed in catalog.

TO:

ECE 30861 Software Engineering, Sem. 1, Class 3, Lab 0, Cr. 3.
Prerequisites: ECE 36800

Introduction to software engineering principles, with special emphasis on the process, methods, and tools needed to develop and test quality software products and systems.

Learning Outcomes: i) an ability to conduct object-oriented design and use unified modeling language. [1,3]; ii) an ability to understand different models of software development processes. [1,2]; iii) an ability to analyze requirements and write project specifications. [1,2]; iv) an ability to successfully develop a team software project on time and meet the specifications. [1, 2, 3, 4]

ELMORE FAMILY SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
Materials Sciences and Electrical Engineering Building, Room 140
501 Northwestern Avenue, West Lafayette, IN 47907
Office: 765-494-3390

Reason: This course has not been offered for 10 years due to not having faculty to teach it; however, students continue to express a high interest in this material. ECE now has faculty members to revise and teach this course with listed updates. This course introduces students to the vocabulary, process, and mindset of a software engineer or related fields (e.g. embedded systems), and provides a team-based software engineering experience. In addition, skills in this course help prepare students for internships as well as full-time positions as software engineers.

Milind Kulkarni
Associate Head of Teaching and Learning
Professor of Electrical and Computer Engineering

Davis – ECE 461 Reboot – Lecture outline

Weeks Major Topics

1	Introduction, project description, team organization.
1	Core concepts in teamwork and tools (project management software, version control, code review, bug tracking, etc.)
1	Introduction to project-relevant engineering infrastructure (web stack, databases, query languages, big-data tools, etc.)
1	Software engineering processes
2	Core software engineering activities: Requirements engineering; Design
1	Midterm project handoffs
2	Core software engineering activities: Validation; Maintenance
1	Software release and maintenance
1	Ethics, reliability, and standards
1	Working with legacy software
1	Working with open-source software
1	Software 2.0 – Incorporating AI & machine learning
1	Project postmortems and presentations

Davis – ECE 461 Reboot – Old vs New

Summary

This is a proposal to re-boot ECE 46100 – Software Engineering.

The course was last offered ~2011. There is substantial student demand, particularly from the (many) CompE students planning to go into software jobs in industry.

The proposal changes the course as follows:

- Change in prereqs
- Different textbook.
- The lecture outline is changed – some reorganization, some new topics swapped in to replace old ones. Details and justification are below.

My goal is to regularly offer undergraduate and graduate courses in Software Engineering. I am currently offering the first edition of ECE 595—Advanced Software Engineering with ~10 undergrads and ~20 graduate students.

Course catalog entry

https://engineering.purdue.edu/ECE/Academics/Undergraduates/UGO/CourseInfo/courseInfo?courseid=402&show=true&type=undergrad

Change in prereq

Old: ECE 30862 (OOP)
Proposed: ECE 368 (Data structures)
Justification:

- **Practical**: ECE 30862 is no longer actively offered; Sam has been teaching ECE 395 variations instead. In contrast, ECE 368 is a "fixed point" in the curriculum.
- **Pedagogical**: The purpose of the prereq is to ensure that students have sufficient programming experience. They should be "fluent" so that they can reason about higher-level engineering concepts. Students who pass ECE 368 have demonstrated this skill.

Textbook

	Existing	Proposed	Why
Required	• Software Engineering: A Practitioner's	• Software Engineering, Sommerville, Pearson 2016.	Pressman vs Sommerville: I have gone over both texts (and several

	Approach. Pressman, McGraw-Hill, 2004.	• Software Engineering at Google, O'Reilly, 2020. (e-edition available through library)	others). I find Sommerville much more readable. SE@Google: Free to access. It gives a deep dive into one company's specific context, instead of the "survey of experiences at many companies" found in most SE textbooks.
Recommended	• Software Design, From Programming to Architecture. Braude, Wiley, 2004.	• The mythical man month: Essays on Software Engineering. Brooks. • Design Patterns (Gof4). • Head First Design Patterns. O'Reilly.	I am not familiar with the Braude book, but students don't need another 600-page textbook. The recommended books cover (1) essays and reflctions on SE; and (2) some specific trouble areas in detail. Design patterns are also treated in the OOP elective (ECE 30862 and its offspring). These are a critical topic and worth covering in multiple places. In ECE 461 the focus will be more on architectural patterns than on class-based patterns.

Lecture outline

Week	Existing	Proposed	Notes
1	Introduction, project description and team organization	*Same*	
2	Requirement analysis and project specification	Introduction to project-relevant engineering infrastructure (survey of web stack, DB, query languages, big-data tools, etc.)	Show students the technical building blocks they need. The undergrad curriculum does not cover most of this. ECE 361 (taught as ECE 495 for the last 2 semesters) did but is being replaced.
3	Version control and bug tracking	Core concepts in teamwork and tools (project management, version control, code review, bug tracking, etc.)	Show students the teamwork tools they will need.
4	Visual programming and user interface	Software engineering processes (waterfall → agile)	Give students vocabulary and define the processes they should be following on their projects
5	OOP and UML	Core activity #1: Requirements engineering	Moved down from week 2
6	Software development process	Core activity #2: Design (OOP & UML)	Moved down from week 5
7	Open-source development model	Midterm project presentations and hand-offs	Open-source covered in week 13. Instead, here, I have the student teams undergo a reorganization. One person will stay on each project and

			the rest will go to another project. This week is an innovation – teach the students about 'bus factor' and how to onboard new team members and adapt to someone else's code.
8	Midterm project presentation	Core activity #3: Validation (Test and verification)	Project presentation was done in the previous week. Validation is moved up from Week 11.
9	Team management	Core activity #4: Maintenance	Team management concepts were covered much earlier, so the students could manage their teams well. Maintenance was somewhat covered in the previous edition.
10	Ethics, reliability, and standard	Software metrics and improvement	Ethics moved to the next week. Metrics & improvement were somewhat covered in old week 14, but this way the students have more time to apply the concepts to their projects.

11	Test and verification	Ethics, reliability, and standards	Test is moved earlier. Ethics is moved a little later.
12	System integration	Working with legacy software	Integration concepts are discussed during Test week. Legacy software is part of a 2-week sequence on "software in the wild".
13	Estimation and product metrics	Working with open-source software	Metrics were discussed in week 10. OSS is part of a 2-week sequence on "software in the wild". Moved down from old Week 7.
14	Software release and post-release analysis	Software 2.0: Incorporating machine learning	Release and analysis concepts are somewhat covered in week week 10. The post-release analysis is handled via a postmortem in week 15. Software 2.0 material is new.
15	Final project presentation	Project presentations and post-mortems	Similar, but incorporates more of the "post-release analysis" aspects.

Questions

"This is an experiential learning course." – What does this mean?