Software Systems

- What are the fundamental differences between hardware and software? Good news? Bad news?
- What is the difference between the problem space and the solution space? Why is this distinction important? What do we do differently in the two spaces?
- Why are theories and models important to software engineering? Can there be multiple theories for the same problem? Can there be multiple models for the same theory?
- How do software systems effect the world? What problems arise from them?

Software Systems

- What is the difference between and E-type system and an S-type program? Why are they important? What is the primary concern of each type? Do S-type programs evolve?
- Why is the distinction between programs as calculations and programs as behaviors important to us as software engineers? Which of the two views is more important to us as software engineers? As computer scientists? (This is a trick question ☺)

Study Questions

- Why is theory, algorithm and data structures important for software systems? What effect does they have on designing, building and evolving our systems?
- Why is experience important? What kinds of experience do we find? How does it affect what we do in designing, building and evolving our systems? What is the ultimate goal of gaining all this experience?
- What are the sources of our methods, processes and techniques? Why are they important? What is their relationship to tools and technologies? How do they affect what we do? How do organizations affect us?

Study Questions

- What is code decay and what are the symptoms of code decay?
- What kinds of metrics can you use to assess software quality?
- How is the “modularity” defined in the paper?
- How is the “fault potential” measured in two different ways in the paper?
- What are two limitations of using statistical regression to examine code decay symptoms in the paper?