Engineering Sermon 1: Simplicity

Motivation

Simplicity must be a central goal in your work (not just a trade-off)

Lots of forces against simplicity

- academia "Wow, you must be really smart to have figured that out"
- academia – complex system → easy to publish incremental changes
- industry – marketing: complexity is good (more features)

I don't buy it! Good science and engineering – make things simpler

- complexity comes from special cases → not a general solution → little overall impact
  - e.g. Maguire "Solid Code": many if statements in a procedure indicates you are trying to make it do too much
- Patterson: best research is obvious in retrospect "Anyone could have thought of that"

Reasons for simplicity

1. Cost of complexity: longer design, construction, test, debug → less impressive results in a fast-moving field
   (also – when you buy into a complex design, you are buying into many many hours in the lab!)

2. can't improve what you can't understand
   fundamental problem with OS's – to fragile to introduce new features found by researchers

   Performance tuning – hard to make big, complex things go fast

3. impact – general ideas have wide influence; complex ideas (less broadly applicable, less widely understood) have little influence

4. Cheaper → quicker to market (or PhD)

How do you make things simpler?

0. Humility – use creativity to simplify, not to show off!
1. Don’t accept complexity. Design then code.
2. Be extremist. Max simplicity.
3. Finish projects
    - Your success is measured by projects you finish, not projects you start
    - Ousterhout: “When I read a paper, I can immediately tell whether the authors ever built anything”
    - all successful projects have a painful middle stage when you get your neat ideas working
    - forcing yourself to finish is how you learn to avoid too complicated ideas