Spring 2020

Introduction

Today

- Intros
- Overview of the Approach
- Logistics (more than usual with remote)
- Class Assumption and Aims + Interactive Portion!
- An Initial Assignment
- Some Intro Ideas woven into Amusements

POWELL CSE 112

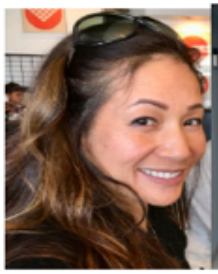
Introduction

Lecturer UCSD CSE Department

Industry CEO, Founder

PINT

Author




Experience
Couchbase
salesforce
Pronouns: He/ Him/ His
Factoid
PES DEBATING SOCIETY
Founded the debating society
Education
PES UNIVERSITY
Undergrad
CSE
Masters
CSE
ML & AI
UNIVERSITY OF CALIFORNIA SAN DIEGO
AMAN ACHPAL

Teaching Assistant

Chen Chen

- 2nd Year Computer Science PhD Student;
- Research Interests: HCI, mHealth, Wearables, AR/MR;
- Pronouns: He/Him/His;
- Educations:

BE EE @UoN (UK)

MS ECE @Carnegie Mellon (PA)

Tutor

Yingzhen(Ada) Qu

Pronouns: She/Her

Education

- Current MS CSE student
- Graduated for BS Fall 2019

UC San Diego

Experience

- OS automation work in Teradata.
- Play Store Commerce team in Google
- Will work on call related features in Google this summer

teradata.

Google

Shardul Saiya

Experience

Campus Mobile App

aira

Managed Team 13 last year

Computer Science (Senior)

Possible Outcome

- Finding out if this really is the class for you?
  - Why it could be?
  - Why it might not be?
- What to do if it isn't and why

From the Past CSE 112s

- I have found that an applied approach to learning the theory & practices of software engineering works best.
- Experience is a great teacher and good experience is generally fraught with risk
- We moved things up front got our soft dynamics better than usual & built the ”build conveyor belt”, yet coders still want to code too early
- We allowed ‘failure’, even **Yoda** says that’s true!

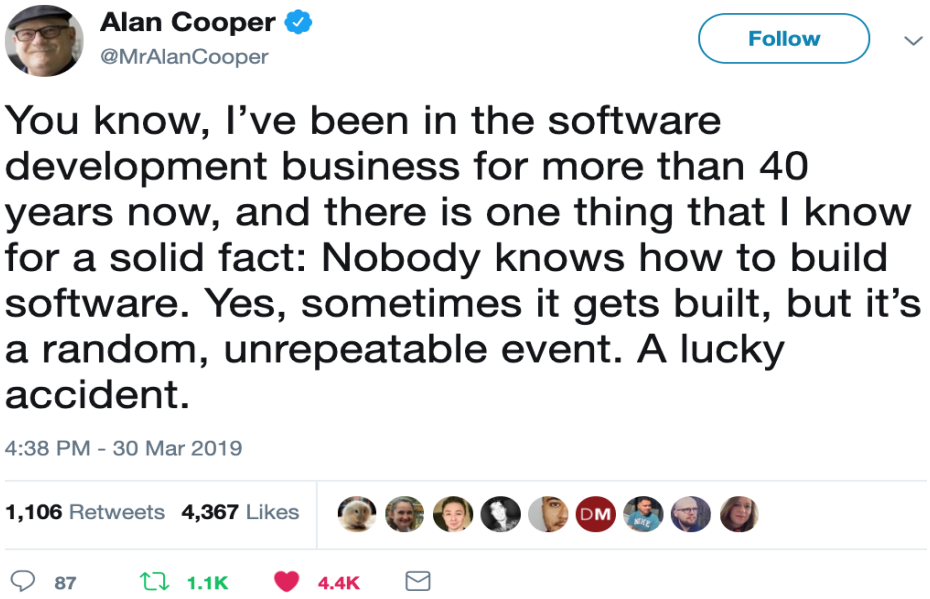
More from before

- Still had some problems, but it gets better each year
- Group issues as always, maybe 2-3 last year
  - We can do even better!
- Even in success a large number of people must experience things to believe this (which is ok)
  - Made it harder for them since it took a while to see other “proof” which put them behind
  - Some disbelief of what I want, which gave me a sad
- Logistics and details always a challenge, this quarter * 100

And even more…

- Even at the end some still didn't see what the course is really about which is frustrating (< 20%)
  - It's more meta - it is about the factory, team and process more than the widget produced.
  - Example: Carpentry isn't just about a particular bird house it is about the process of making a bird house and the associated skills
- Lot's of same old buried beyond artifice
  - The crunch time, the handwave, the demo illusion

Cranky Old Man?



Our Construct: Peter Venkman

- First year, Peter the Entrepreneurial Frankenstein with small teams
- Second year, Peter tries to crowd source his success with two 60+ teams
- Third year, Peter returned to tempt teams of 10-12 with new and old code
- Fourth year, Peter gets the software factory going and things are better, but many teams resist until nearly too late
- Fifth time, Software factory is up early and less failure than before but scratch deeper and see trouble “Hero units” , ”fake until make it”, etc. and really lacking delivering complete value still
- Sixth time, factory up early, even less fail, Peter less important, but now feeling we aren’t honest about shipping and the sameness of any group build with skills

This time in CSE 112

World: Global Pandemic & Economic Crash

UCSD: Full remote teaching for a heavy in-person style class!

112 Prof & Students

Yet before 112, we have 110

- Need to collect some data about 110 and what you come with for me to understand the best way to teach this class

Preview of Our Project

- Process over Project
- Work up front, but iterative and …
- Constant Effort (Starting today even!)
- Big project made small
- Relatively Fixed Tool Chain
- Up next the actual premise

The Premise

- Simulated “life” approaching as opposed to PA style life
- Normally we used a client construct this quarter we use a situational construct
- “Pandemic” means we are all remote first!
- We don’t have the tools nor the techniques for this, so why grumble let’s make them!

The Premise Contd.

- Limit tool and task switch.
- Address attention and discipline
- Address process and communication
- Fixed duration, variable scope!
- Ship Quality or Ship Nothing

Three Phases+

- Tutorial Level: Teams, Tool/Process Prep + Green Field Code to get the "belt" right
- Normally
  - Level 1: First Component
  - Level 2.1 – 2.N: N Components
  - Level 3: Class Demo
- This Remote Shortened Time
  - Prep & Initial Practice (~3 week)
    - Potentially internal "belt" competition
    - Ending in internal pitch
  - 6-week Work Phase on accepted pitch (your or others)
    - Potentially internal subphases
  - Shipping and External Presentations

Logistics

- Many many many (data) points
- Sure there will be a midterm & typical ~4 individual pieces
- Then there is the project and process
  - Points for attendance, daily check-ins, weekly sprint meeting, weekly TA/tutor meeting, 2 prof meetings, code review proof, presentations, on and on and on …

Project or Process

- This is a process class with a project not a project class with a process
- Weight of work is more up front if you are on track for a great grade. We start literally today
- There is a time commitment here, but it is hopefully worth it and is directly applicable to your careers

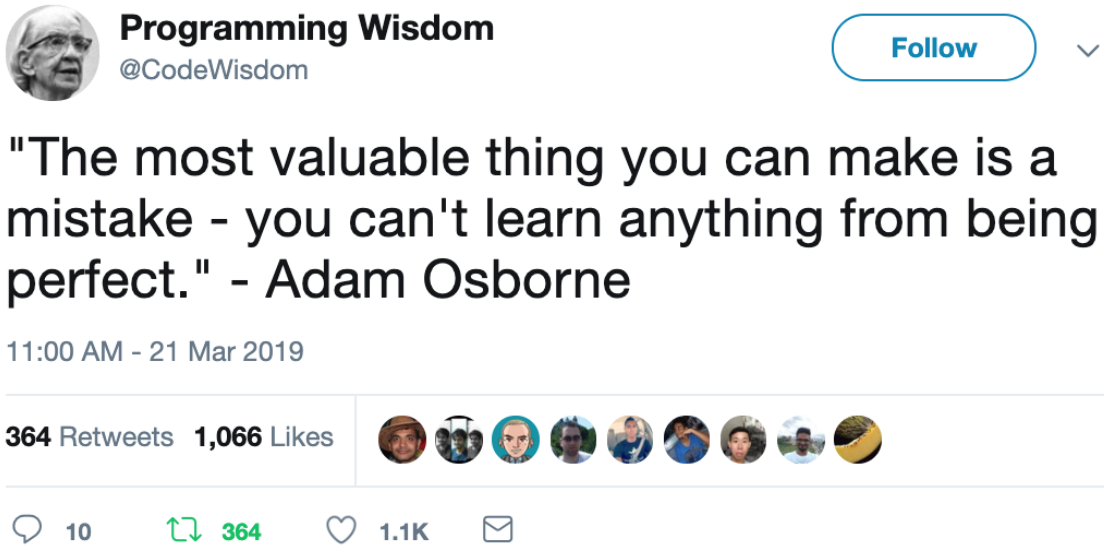
More Logistics

- From the previous students many ideas
  - Trying to eliminate more group schedule concerns
  - The discussion section solution
  - The major penalty suggestion
  - Applications for leadership
  - Team selection by skill balance
  - The importance of the team build and forcing the communication
  - The check-in and effort requirements
- All this might be troubled by “remote first”

Teams

- Assigned with some logistics (time zone), balance, and previous collaboration consideration
- Size: ~11-12 may fluctuate for obvious reasons
  - 2 Comms & Lead
  - 2-3 Coders
  - 2 Quality.
  - 2 Build
  - 2 Research / Design
  - up to 2 Wait Listers
- Survey will help, but first step is lead selection and then teams will be formed

Win with a Fail?



“Real World” Grading?

- Matrix grading?
- Dimensions
  - Work Effort
  - Process Adherence
  - Success - Shipped Code (or not)
  - Quality
  - X Factor / Happiness?

Grading FAQs

- You can “fail” the project and excel in the class
- Being in a group is a group commitment there will be grade variation, but it will be tough to be a grade superstar in a failing group
- Being the leader is not a grade bonus guarantee, leaders have been fired in previous class iterations
- Status and points will be pretty transparent, but like industry subjective measures will be more than a minor aspect of evaluation. Advice given now if that bothers you.
- UCSD current quarter attitude about grading

My Assumptions

- You know how to program
- You have taken an intro course in software engineering
- You have a reason you want to be here
  - There is some expected outcome I hope beyond the grade and units if I we both do our jobs right
- Some of you will have worked (internship, part time, etc.)

My Class Goal

To help students gain some initial practical working knowledge of how to produce software in an "engineered" fashion balancing theoretical aspirations with the practicalities imposed by the development environment that will be encountered out in the "real world". I also aim to do this is a focused fashion that is deliberate and iterative and move the ownership of learning to where in belongs.

Step 1: Watch

Step 2: Think Soccer?

- What are the steps as it goes?
  - Team formation, players, positions, practice, etc.
- Raise your hand in Zoom now let's discuss for a few minutes the linear progress of being an elite player

Step 3: Think Coaching

- So did ~~sports~~ teams need coaches / trainers?
- Should I be the coach of this?
- Arguments for me as coach
  - Experience, success, team size, position, credibility, etc.
- Arguments against
  - Age? Degree of success? Specific technical awareness?

Your Beliefs

- Come crashing into my goals and biases
  - Many of you really already have strong beliefs based upon what you think software engineering is or should be
- We must get those out in the open for a variety of reasons if we are to succeed
- Yet I don't need you to parrot my beliefs I need you to be firm and well founded in YOUR beliefs
  - It will your career to grow not mine! I just coach you at this early step.

Ultimately it comes down to this when it comes to skills and beliefs

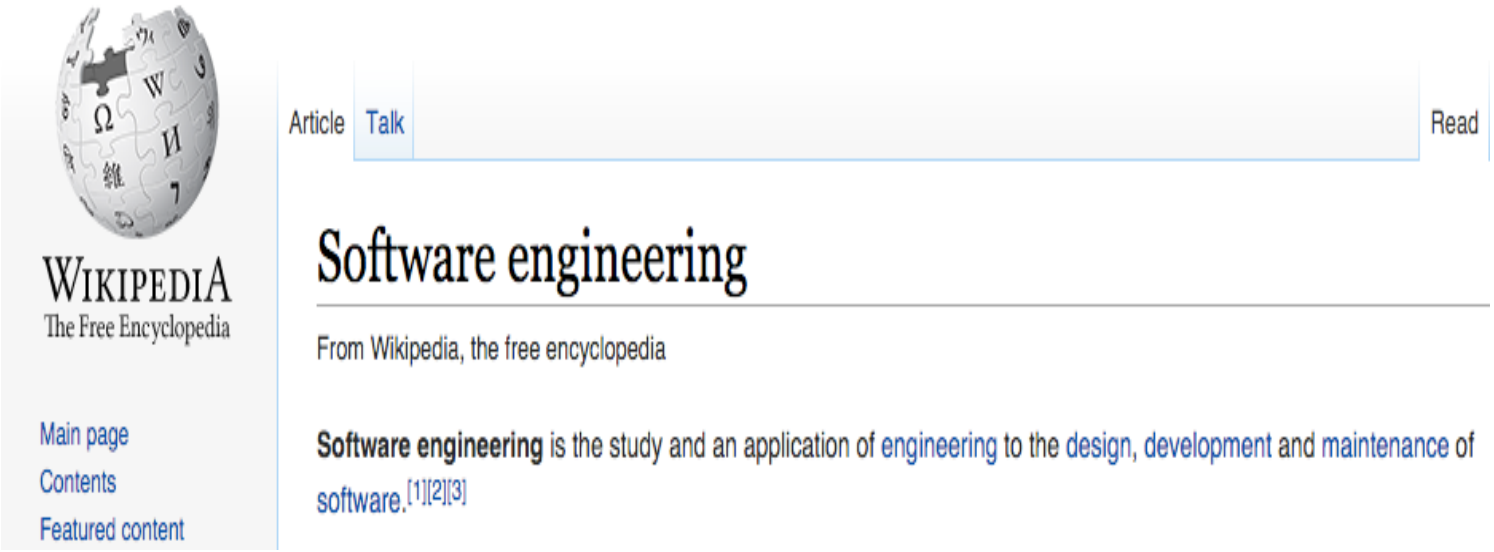
Ok sure sure, so SE what is it?

Yeah you already took a class in it?

What is is this topic all about?

3 brave people raise hand please!

What is Software Engineering?



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content

Article Talk Read

Software engineering

From Wikipedia, the free encyclopedia

Software engineering is the study and an application of engineering to the design, development and maintenance of software.[1][2][3]

Getting more precise…

Typical formal definitions of **software engineering** are:

- "the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software";[5]
- "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software";[6]
- "an engineering discipline that is concerned with all aspects of software production";[7]
- and "the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines."[8]

Operative Words

Typical formal definitions of **software engineering** are:

- "the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software";[5]
- "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software";[6]
- "an engineering discipline that is concerned with all aspects of software production";[7]
- and "the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines."[8]

Engineering!

- *Engineering* is derived from the Latin *ingenium*, meaning "cleverness" and *ingeniare*, meaning "to contrive, devise"

  *"The **creative application of scientific principles to design or develop** structures, machines, apparatus, or manufacturing processes, or works utilizing them singly or in combination; or to construct or operate the same with **full cognizance** of their design; or to **forecast their behavior under specific operating conditions**; all as respects an **intended function, economics** of operation or **safety** to life and property"*

What!?

http://www.theatlantic.com/technology/archive/2015/11/programmers-should-not-call-themselves-engineers/414271/

Homework #1 – Due Thursday

First - Carefully read the Atlantic article

Second – Stop and THINK

Third - write a single page document that indicates what we should do to become software engineers in the true sense of the world. This should include personal and field wise thinking (field wise means as an industry) attempt.

Homework #1 Format

In the upper corner include a picture (headshot) of yourself and your name with the statement. ”

I __*yourname*___ intend to become a software engineer” to that end I need to:

Next present bullets that start with things like “Learn…”, “Always”, ”Aspire to …

Which are you ideas about what we want to accomplish as SEs. This is not an assignment to debunk the point of the article. Assume they are correct that we aren’t engineers…yet BUT we want to become one!

Logistics Review

- Decide if this sounds about right for your or not
- Sign-up for Slack if not already there
- Fill in survey posted on Canvas
- Work on HW1 and turn in as PDF via Gradescope
- Get ready for an interesting quarter seeing if remote first works that well!