A Topics Course in Empirical Software Engineering: Bridging Research and Practice

Your class will start soon!
Make sure you have Slack open!

University of Victoria

A Topics Course in Empirical Software Engineering: Bridging Research and Practice

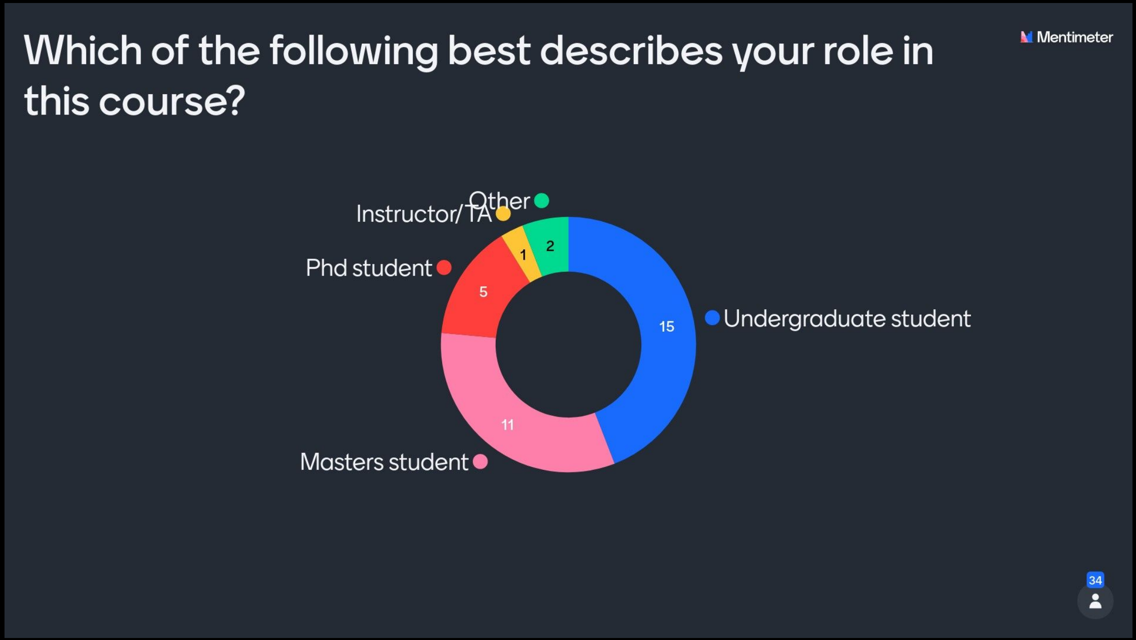
Week 1, Sept 11 2020

Instructor: Margaret-Anne (Peggy) Storey

University of Victoria

Let's start with a quiz!

https://www.menti.com (results are posted under resources!)



Which of the following best describes your role in this course?

Software is eating the world…. Marc Andreessen

https://a16z.com/2011/08/20/why-software-is-eating-the-world/

https://medium.com/@ChrisHerd/why-remote-work-is-eating-the-world-9b773436043a

(Competing) concerns in software engineering…

Code: faster, cheaper, more features, more reliable/secure

Developers: more productive, more skilled, happier, better connected

Organizations/communities: attract/retain contributors, encourage a participatory culture, increase value

Some questions practitioners may care about….

What is a good architecture to solve problem x? [Devanbu]

What makes a really awesome programmer? [Software managers]

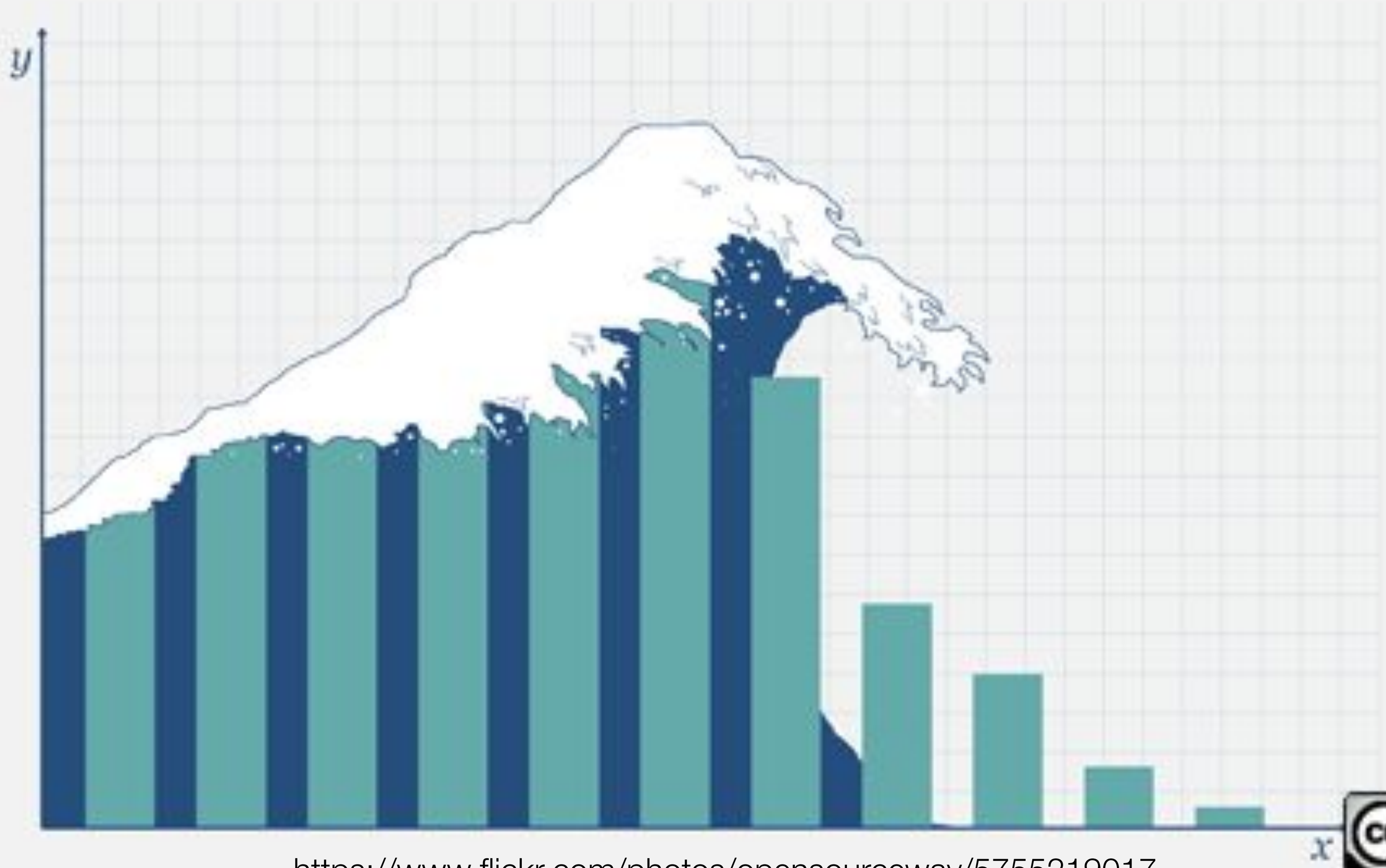
How to build a great development team? [Google]

How is program knowledge distributed? [Naur]

What is the ideal software engineering process?
[Facebook, Microsoft, IBM,…]

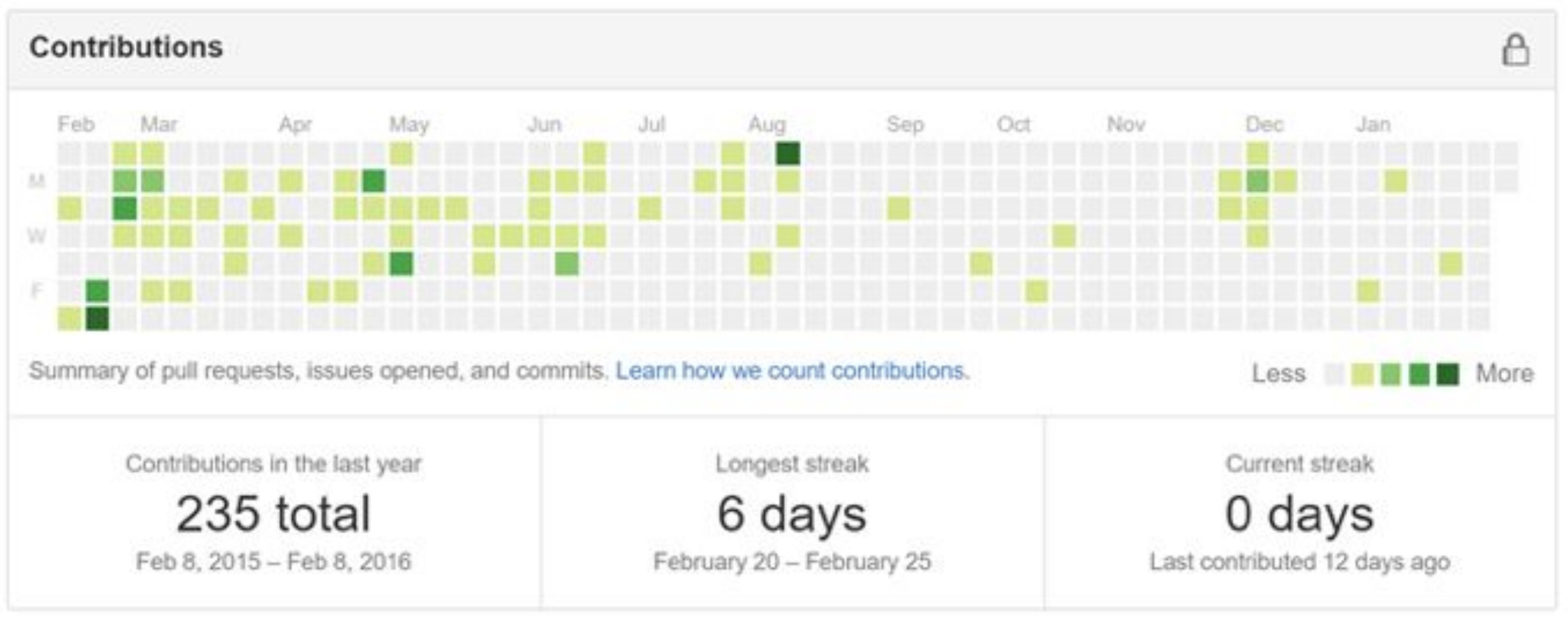
What tools/practices support a participatory development process? [Storey et al.]

Do the answers lie in here?



https://www.flickr.com/photos/opensourceway/5755219017

“Measuring programming progress by lines of code is like measuring aircraft building progress by weight.”



Contributions

Summary of pull requests, issues opened, and commits. Learn how we count contributions.

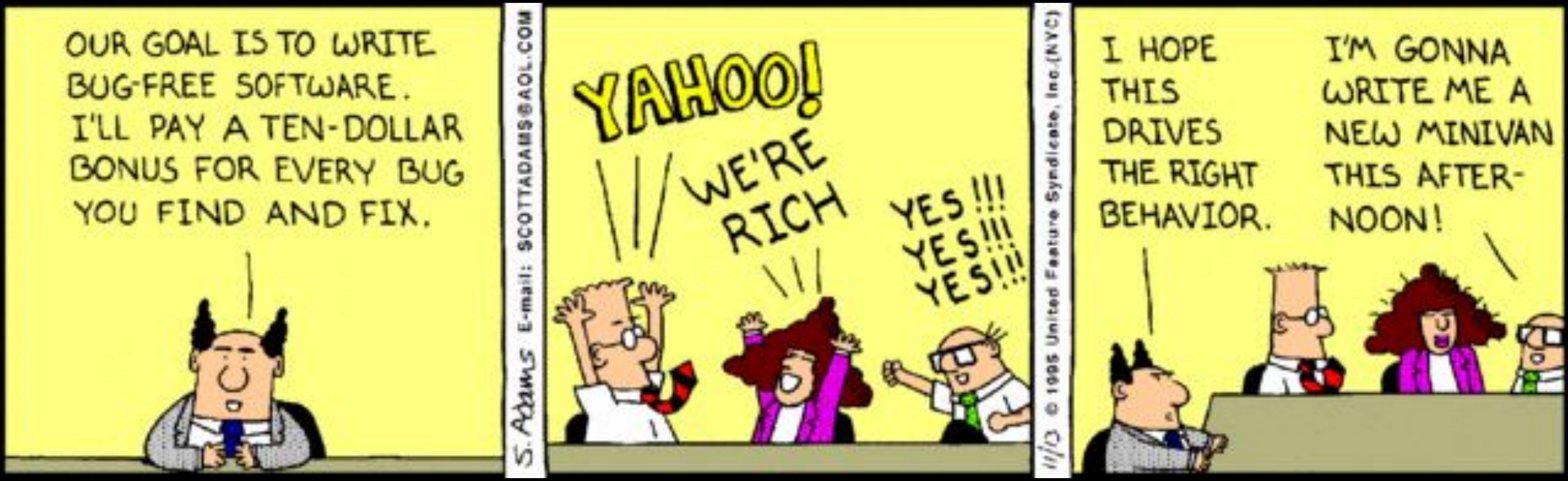
Less ⬜🟩🟩🟩🟩 More

Contributions in the last year
235 total
Feb 8, 2015 – Feb 8, 2016

Longest streak
6 days
February 20 – February 25

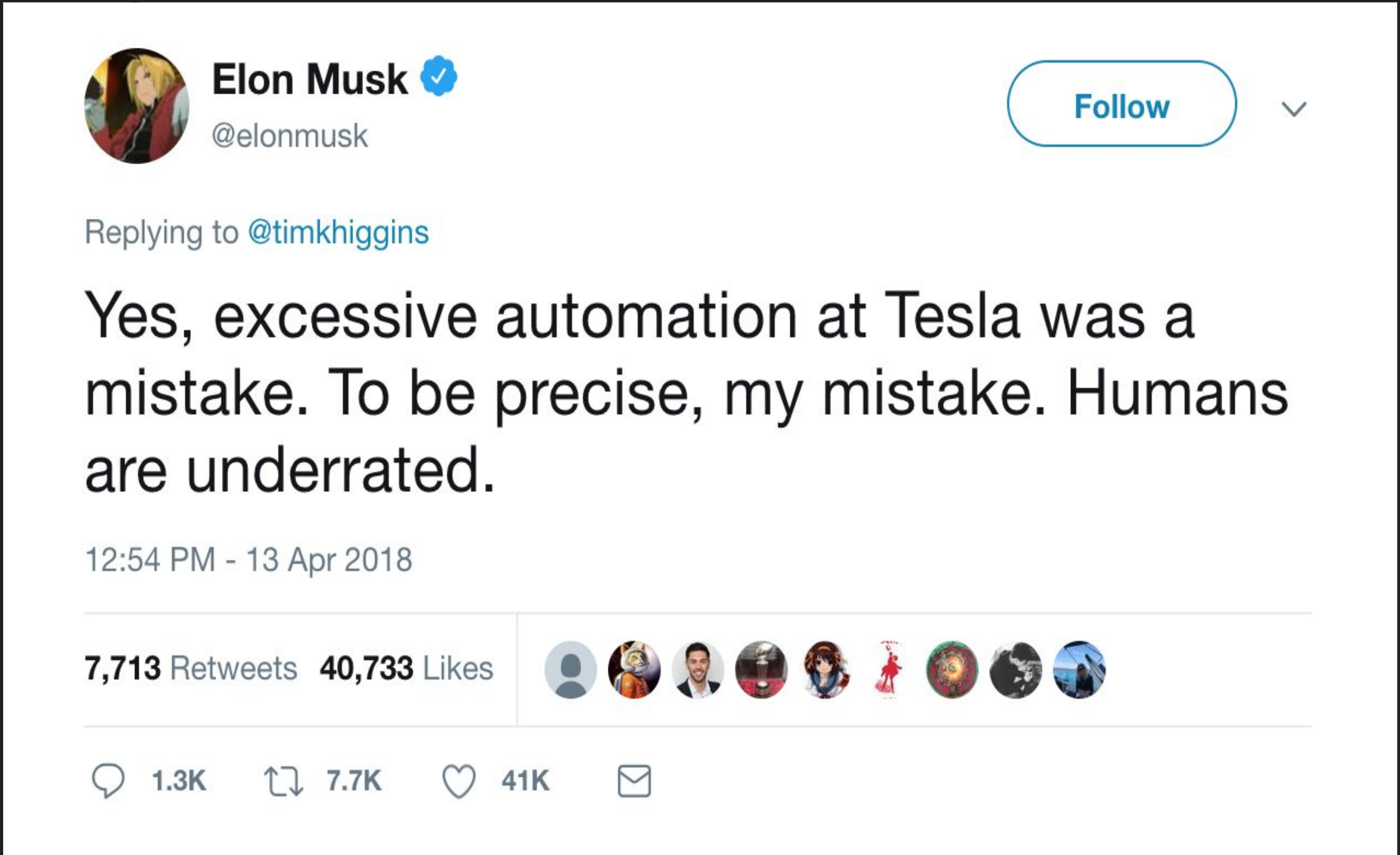
Current streak
0 days
Last contributed 12 days ago

Contributing graphs considered harmful (Hanselman)


OUR GOAL IS TO WRITE BUG-FREE SOFTWARE. I'LL PAY A TEN-DOLLAR BONUS FOR EVERY BUG YOU FIND AND FIX.
YAHOO!
WE'RE RICH
YES!!! YES!!! YES!!!
I HOPE THIS DRIVES THE RIGHT BEHAVIOR.
I'M GONNA WRITE ME A NEW MINIVAN THIS AFTERNOON!

S. Adams E-mail: SCOTTADAMS@AOL.COM

11/13 © 1995 United Feature Syndicate, Inc.(NYC)



Elon Musk
@elonmusk

Replying to @timkhiggins

Yes, excessive automation at Tesla was a mistake. To be precise, my mistake. Humans are underrated.

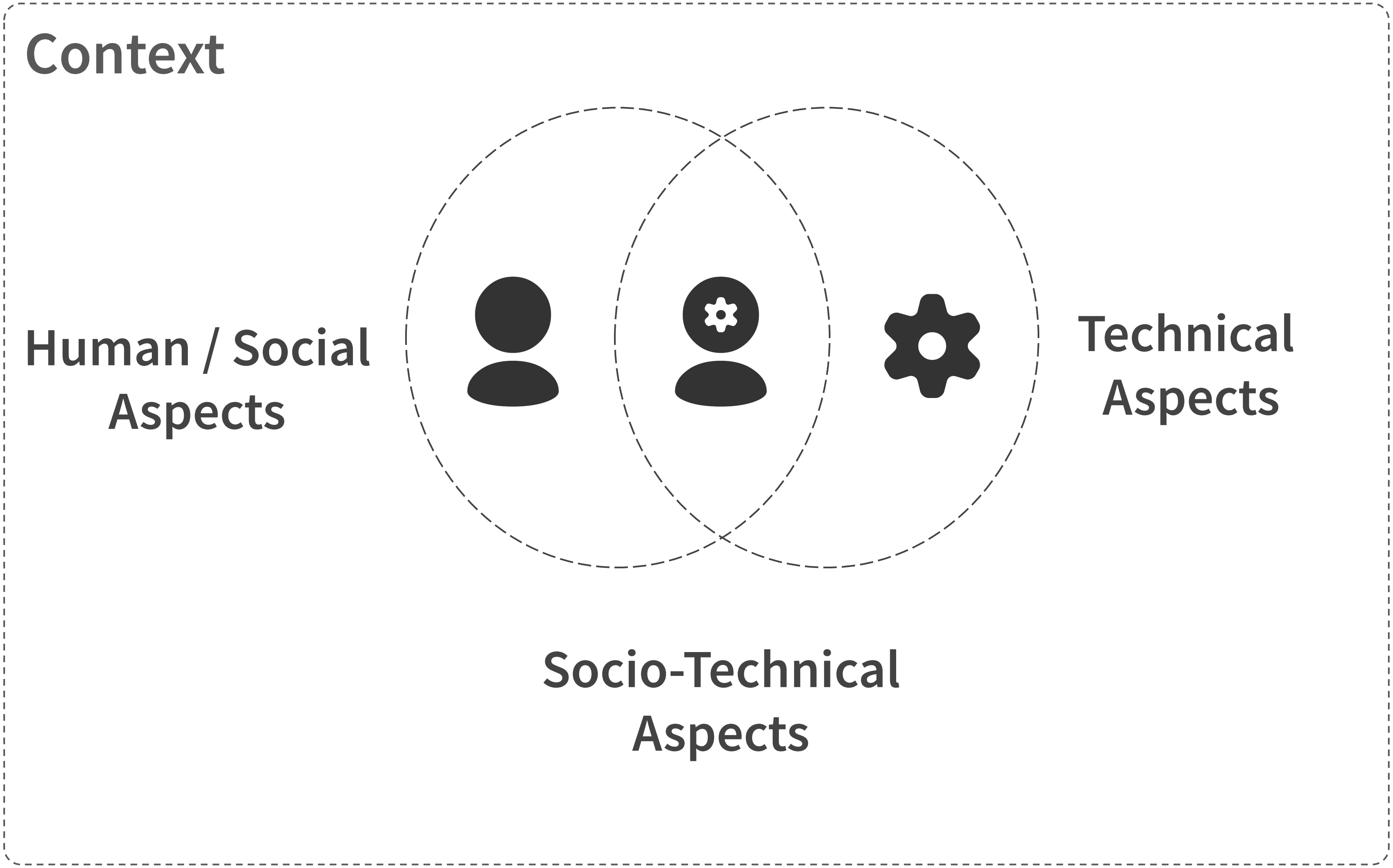
12:54 PM - 13 Apr 2018

7,713 Retweets **40,733** Likes

stackoverflow

#1	#2	#3
41.8%	**36.6%**	**36.5%**
Distracting work Environment	Meetings	Non-development work

Developer Study



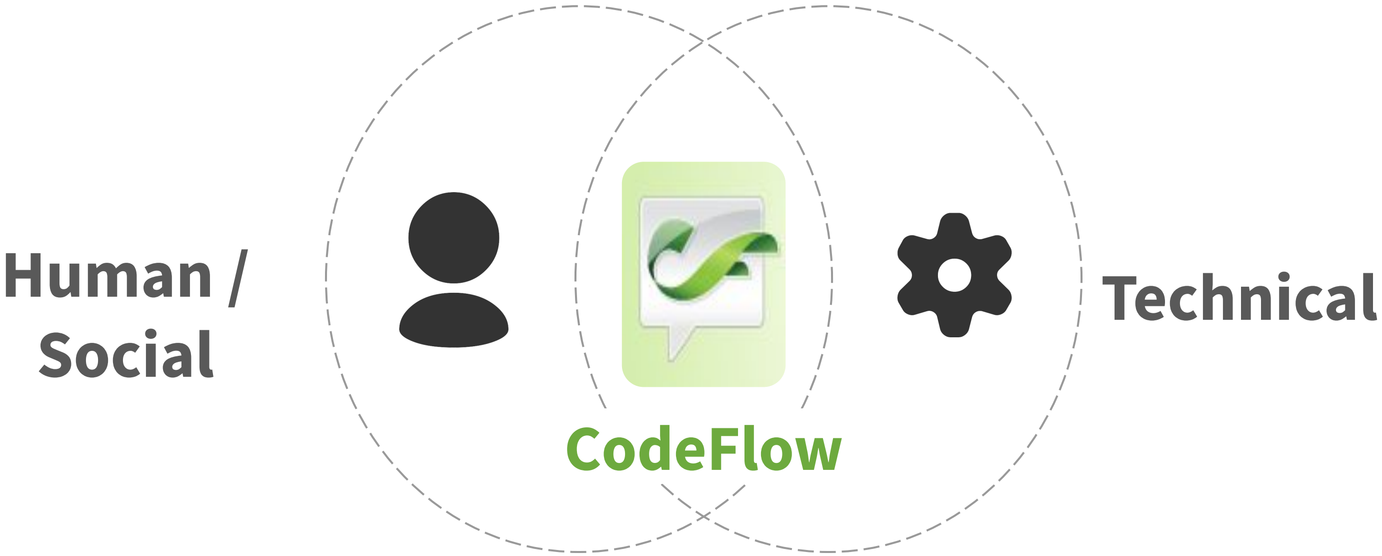
Context

Human / Social Aspects

Technical Aspects

Socio-Technical Aspects

Software Engineering **Design** Space



Human / Social

CodeFlow

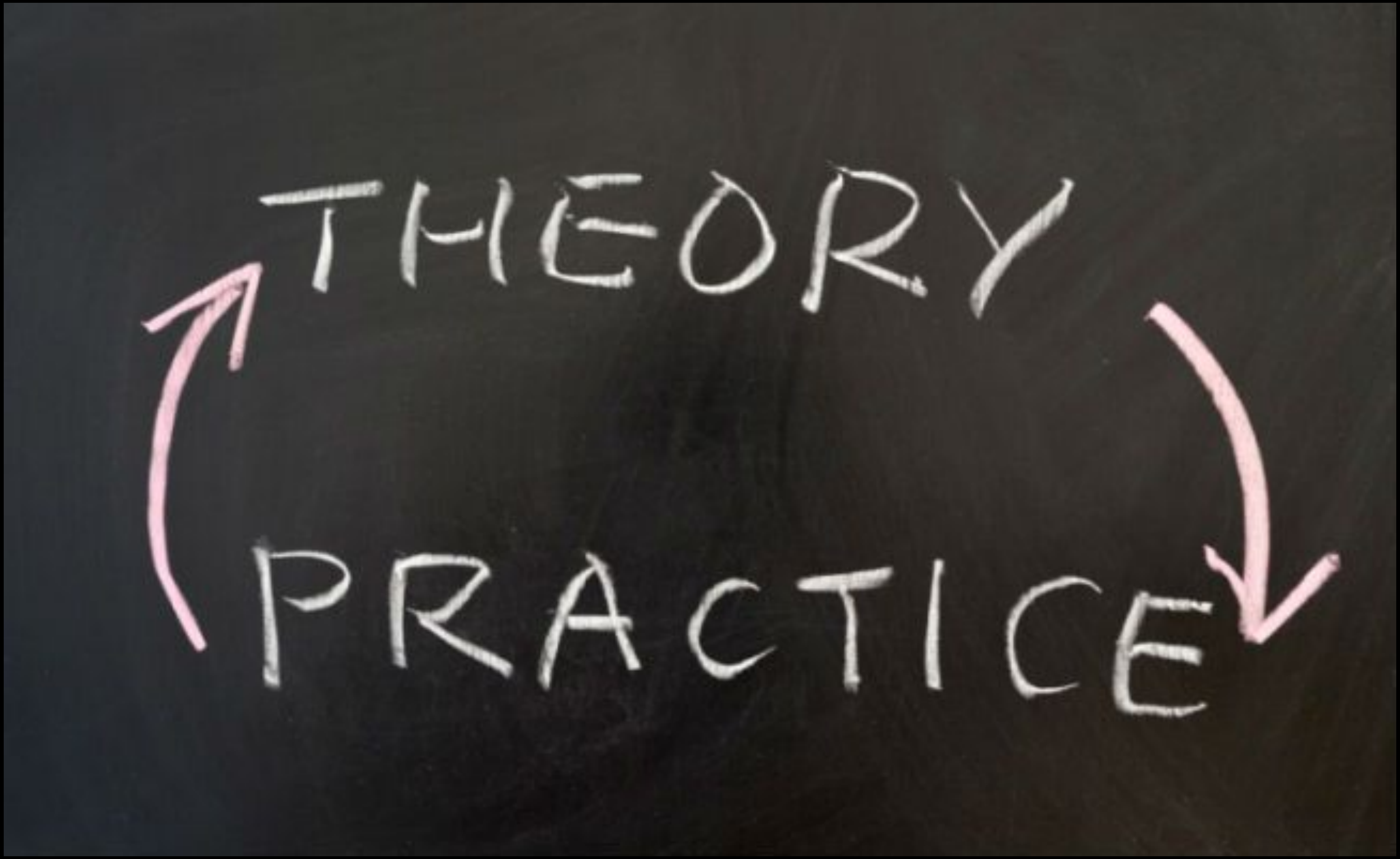
Technical

Socio-Technical

Joint Optimization – Code Review

CodeFlow: Improving the Code Review Process at Microsoft, Czerwonka et al. 2018.

Research success?


THEORY
PRACTICE

Dispelling myths in software engineering (or creating new ones?)

Does increasing code coverage of testing reduce bugs? No, wasting time testing simple code may increase the presence of bugs! [Mockus et al.]

Test driven development reduces bugs, but increases time delivering code [Nagappan et al.]

Geographical distance doesn’t matter much [Bird et al.]

Code clones do not reduce quality in code [Rahman et al.]

References for previous slide

A. Mockus, N. Nagappan, and T. Dinh-Trong, “Test coverage and post-verification defects: A multiple case study,” in ESEM, 2009, pp. 291–301. (note see also this reference for a more recent paper on this! https://ink.library.smu.edu.sg/cgi/viewcontent.cgi?article=4915&context=sis_research)

Nagappan, N., Maximilien, E.M., Bhat, T. et al. Realizing quality improvement through test driven development: results and experiences of four industrial teams. Empir Software Eng 13, 289–302.

C. Bird, N. Nagappan, P. Devanbu, H. Gall and B. Murphy, "Does distributed development affect software quality?: an empirical case study of windows vista", Communications of the ACM, vol. 52, no. 8, pp. 85-93, 2009.

M. S. Rahman and C. K. Roy, "On the Relationships Between Stability and Bug-Proneness of Code Clones: An Empirical Study," 2017 IEEE 17th International Working Conference on Source Code Analysis and Manipulation (SCAM), Shanghai, 2017, pp. 131-140.

Success practice transfer stories from research

Automated testing (Facebook)

Code review tools (Microsoft)

Software Analytics (Hassan et al.)

….

“Academic software engineering research is a backwater with a tenuous connection to practical software development”, Derek Jones

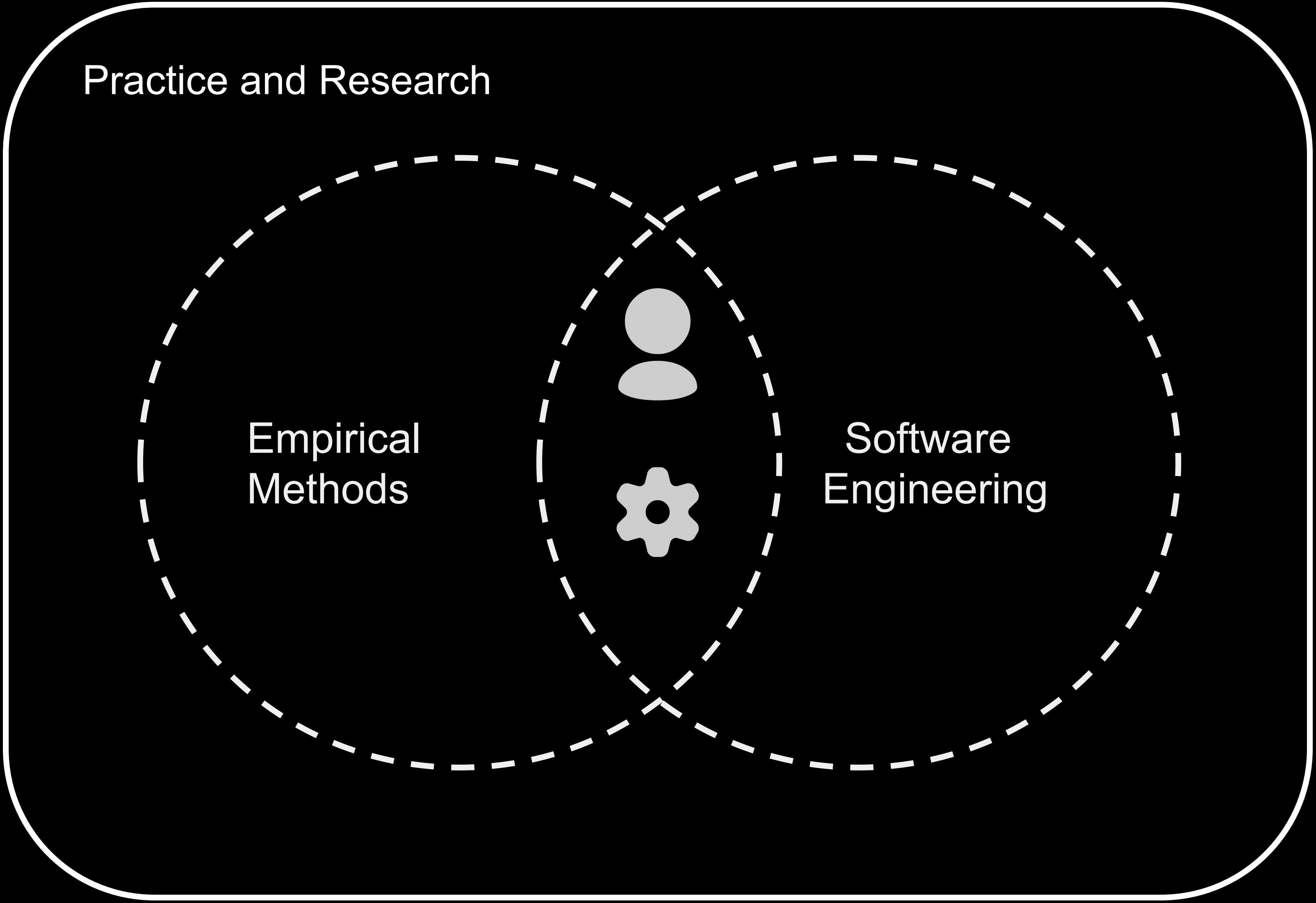
Lack of **industrial relevance** (doesn’t scale or solve industry problems) [Briand]

Poor replication of software engineering studies [Menzies et al.]

Poor actionability (practitioners know which modules are buggy…)

Perils of mining software repositories [Kaliamvakou, German et al.]

Lack of focus on **human/social aspects** [Storey et al.]



Practice and Research

Empirical Methods

Software Engineering

Some of the course objectives….

Appreciate that practitioners are researchers

Learn how to improve the reliability of and be able to critique research in software engineering

Explore how to do research that is more practically relevant

Objectives for today

Course structure (schedule, tools, participation, assessment...)

Group Activity (Zoom Breakouts)

- Mindmap/timeline of History of Software Engineering
- (Video on History of Software Engineering)

Margaret-Anne (Peggy) Storey

- Professor of Computer at University of Victoria
- Canada Research Chair in Human Aspects of Software Engineering
- Director of the Matrix Institute, UVic (Applied Data Science)
- Professor of Aeronautics/Astronautics at MIT n 2000/2001
- Visiting Professor, Lund University (2016-2018)
- Consultant for Microsoft for the past few years
- Faculty Scientist with IBM for over 10 years
- Consulted/collaborated with small companies, organizations such as the World Health Organization, DRDC Canada
- Research interests: developer productivity, development tools and processes, research methods, collaboration
- Personal interests: Sailing, music, grew up in Ireland

Omar (your TA)

“I am a software engineering researcher who studies how automation, and the practices built around it, impact software developers and their development workflow. I investigate the socio-technical systems facilitated by automating aspects of the software development process in an attempt to improve existing automation-enabled workflows as well as explore new ones.

I am also a software developer experienced in several programming languages, frameworks, and paradigms. I am also well-versed in continuous integration tools, container-based development, and infrastructure configuration management. In addition to my development experience, I have experience in the data science field given the research I do, and I am also experienced with machine learning models and development.

Course outline

Same for all students but deliverables/expectations differ for graduates and undergraduates

Note: **Class attendance/participation is required** to pass the course (see the other policies)

https://heat.csc.uvic.ca/coview/course/2020091/CSC578A

Course structure

Part 1: Fundamentals

Lectures, activities

Guest lectures/interviews

A lot of reading outside of class (to lay the groundwork for projects and activities, may be more for graduate students)

Project 1

Part 2: Workshops: deeper dives

Some lectures, activities

More guests

Additional readings

Project 1/2

Tools we will use in this course

Zoom (will record but not post most of the lectures)

Github (schedule, links to materials, slides)

Slack (back channel during class, group discussions, activities, 1-1 direct messaging)

Connex (for submitting some project deliverables, grades, student list management, emergency emails)

Wordpress (will use later for workshop blogposts)

Other tools as needed!

Copyright

Class (zoom) etiquette

You have to be authenticated to enter Zoom

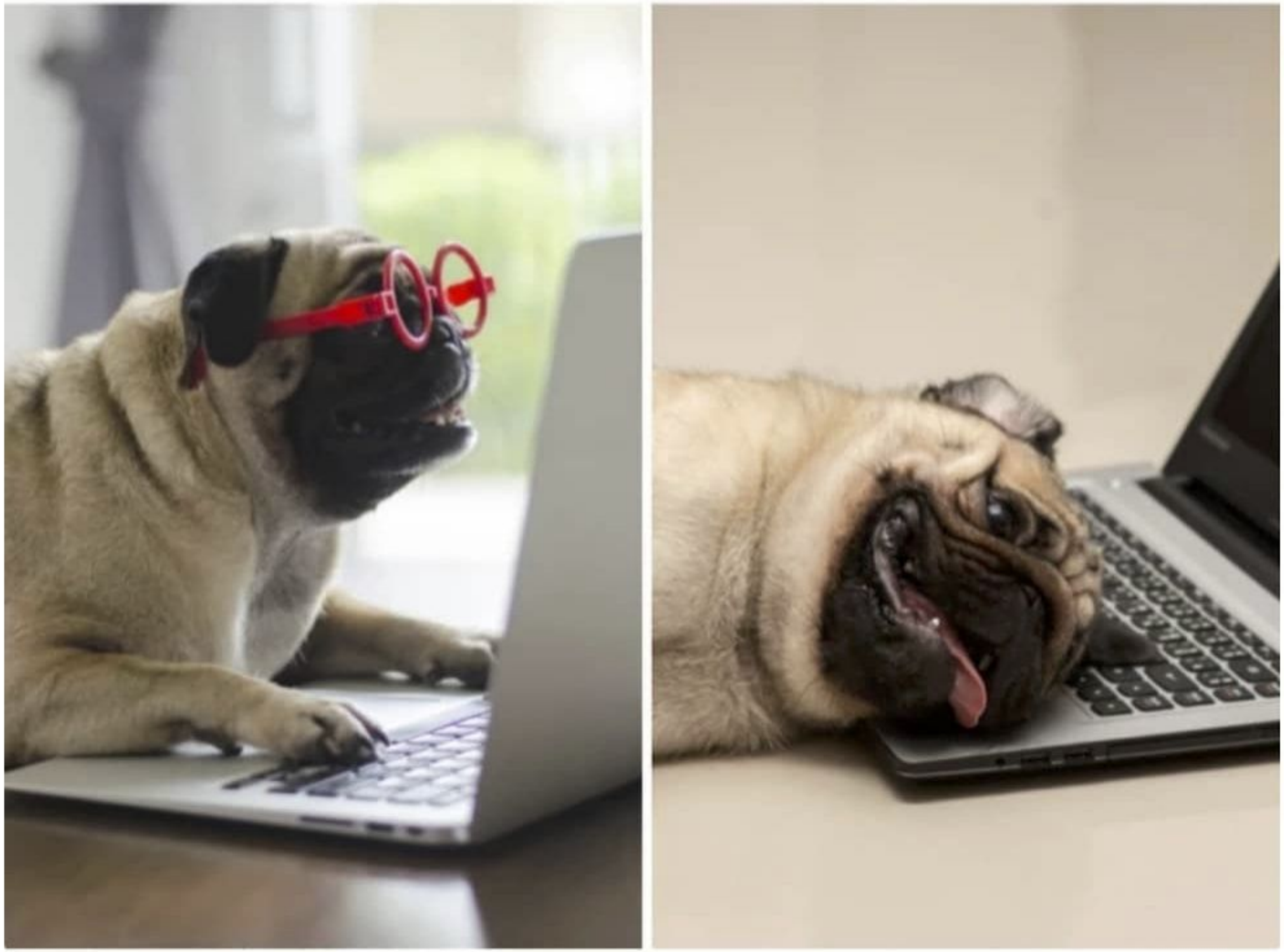
If you want to ask questions or make comments, use your microphone (or raise your hand)

Encouraged to post comments in Slack

Arrive on time

You may not record the lectures (even for personal use)

Add your picture to GitHub, Slack and Zoom (camera use encouraged!)



Zoom meeting, with video

Zoom meeting, audio only

Zoom Meeting

Audio only

With video

When the teacher calls on you during Zoom and you pretend that the connection is bad

Avoiding Zoom Burnout!

Stand/stretch frequently

Breaks (remind me if I forget!)

Drink water

Try to change location in your room or apartment

Two Projects

First project will be an individual literature review (with a twist) and a collaborative component

Second project will be the design (or critique/redesign) of an empirical study (in teams)

Different requirements for undergraduate and graduate students

Workshops: Deep dives on selected topics

Invited speakers (videos, podcasts…), Activities

Topics include:

- Code assessment techniques
- Knowledge sharing and collaboration
- Developer productivity and remote work
- Continuous software engineering (devops, security)
- Diversity and inclusion in software engineering
- Ethics and reliability of SE research

Next week! Introduction to Empirical Software Engineering

Invited guest: Dr. Greg Wilson (please have some questions ready, see GitHub for readings)

Preparing for class: See the readings/podcast in GitHub

Next up today!

Break (stand, drink water, get some fresh air)

But first -- Breakout Rooms/groups activity!

Sketch a timeline/mindmap of the history of software engineering using any tool you like

Watch the video by Grady Booch (see GitHub)

Submit via Slack (put names on your submissions!!!)