On Groups

Lecture 3

How to join

Web

1

2

Text

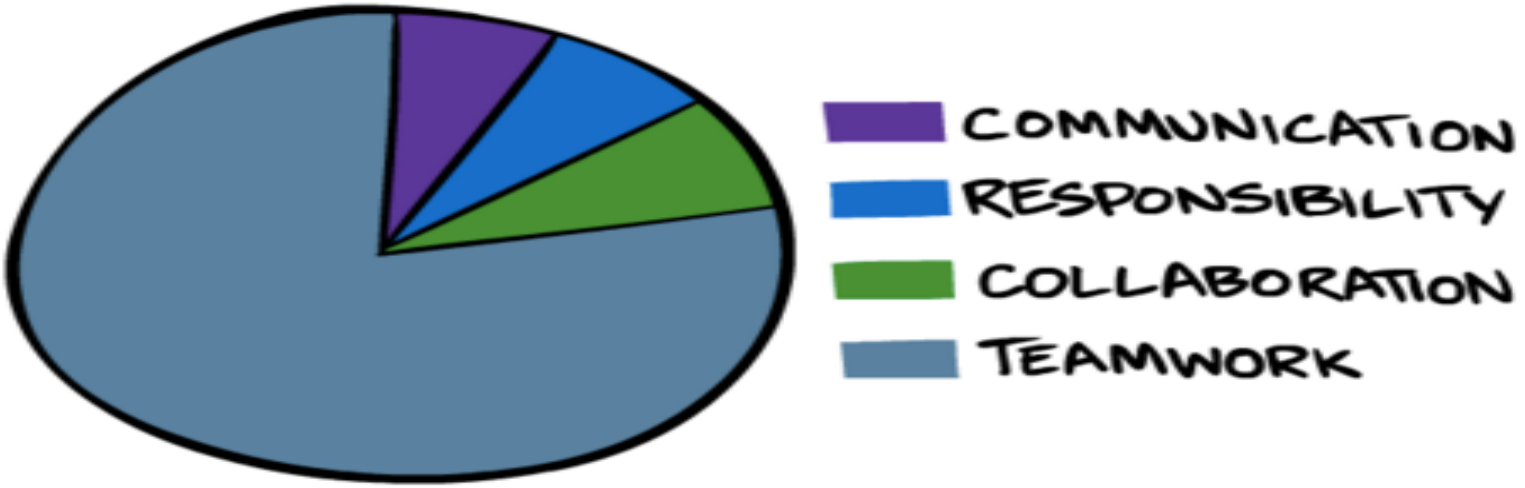
1

2

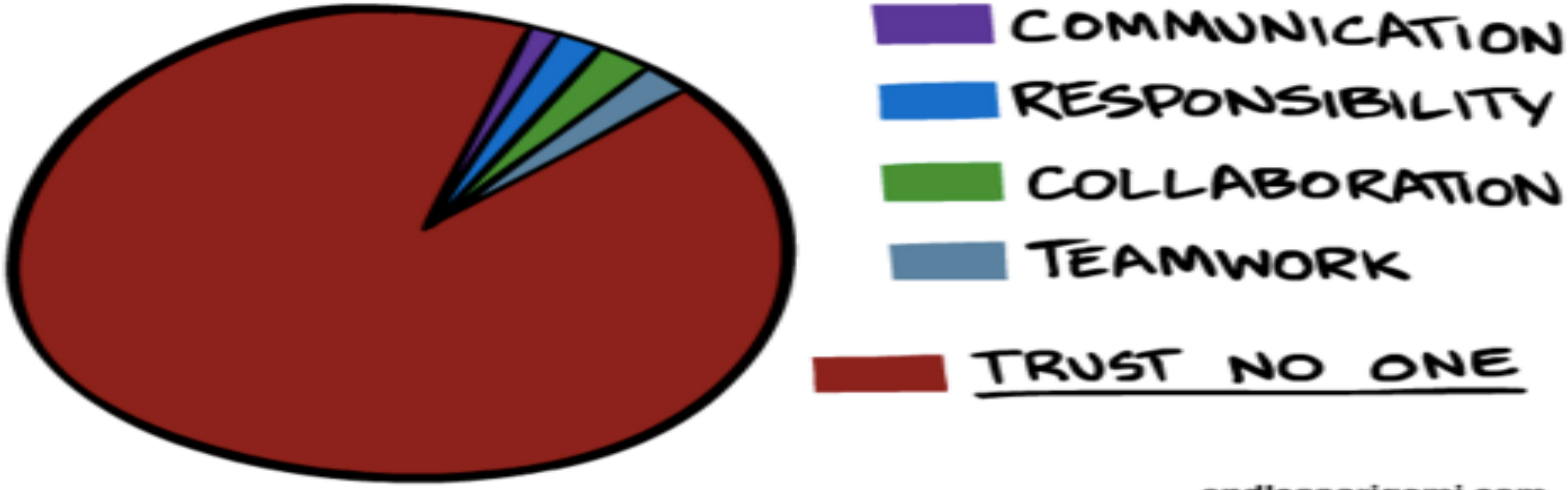
Things to avoid to doing as they might lead to a bad team

Things that could be done that would make a good team

WHAT GROUP PROJECTS ARE SUPPOSED TO TEACH YOU



WHAT GROUP PROJECTS TAUGHT ME



The Sad Reality of Group Projects

- Reality: Forced Collaboration

- Positive Point: Test your metal, learn from mistakes
- Negative Point: Unprepared for the experience to likely succeed

- Truthfully, the Professors' have a hand in this too as we rarely give you any significant amount of coaching on how to succeed

Stages of Group Development

- Tuckman (1965) defined four stages that a group make go thru in development. A fifth is commonly added

1. Forming
2. Storming
3. Norming
4. Performing
5. Adjourning (the wind-up step)

Note: There really isn't a set number of name of steps and numerous models exist (https://en.wikipedia.org/wiki/Group_development) a key point is just to except that a group evolves and should be self-aware if it is to become high functioning

https://en.wikipedia.org/wiki/Group_development
https://en.wikipedia.org/wiki/Team_building

Clearly Don't Do This

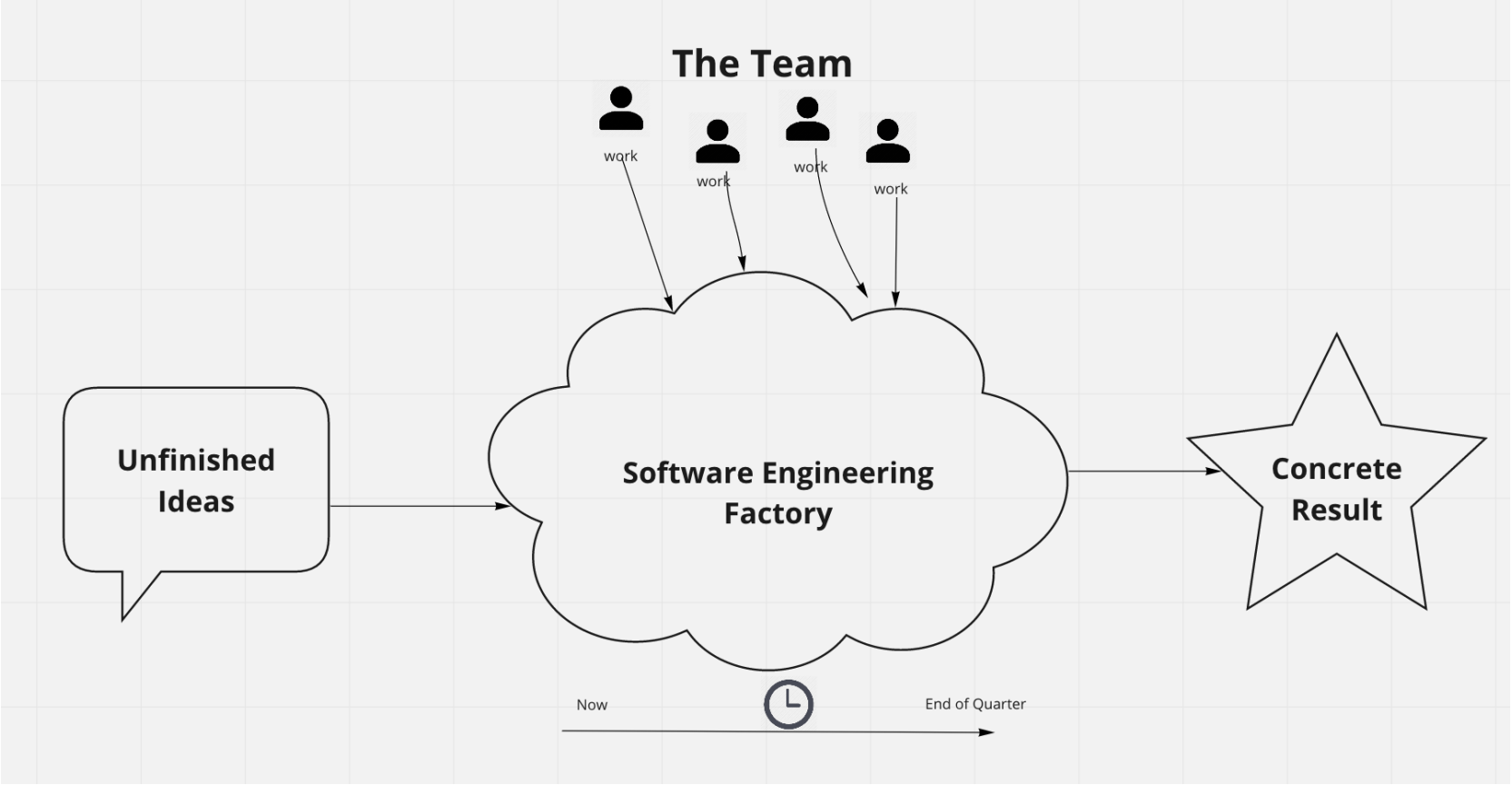
Culture is

Mostly emergent

Mostly defined

Initially defined, but ultimately emergent

The Game For Our Team



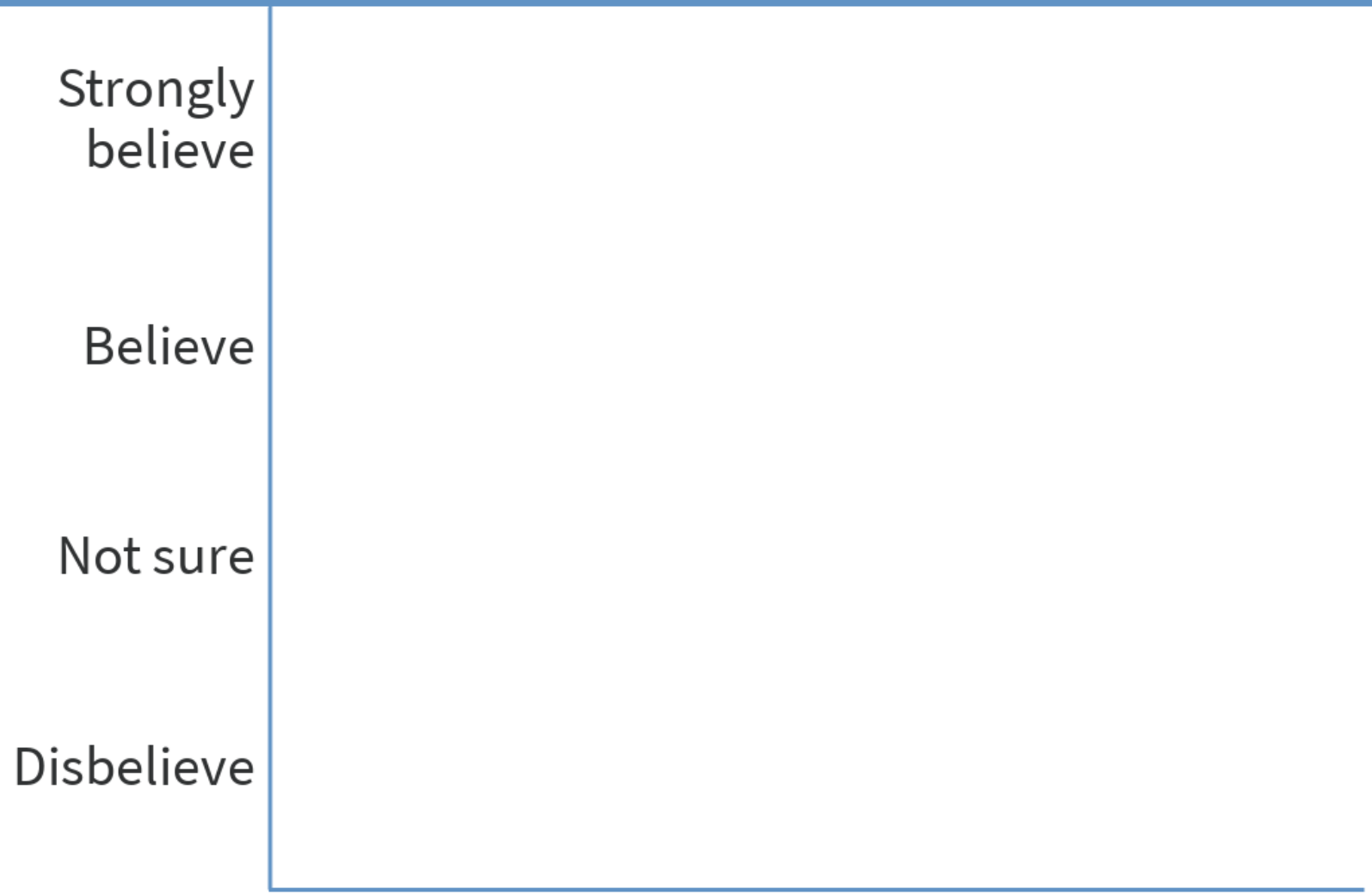
Is This Our Process?

- Step 1 – Make a team
- Step 2 – Somebody Tells Us What to Do
- Step 3 – We code like crazy
- Step 4 – Release
- Step 5 – Profit

- Clearly that is ridiculous but even if was about right step 3 is about more than just coding it requires coordination and communication more than anything

- Communication isn't that easy!

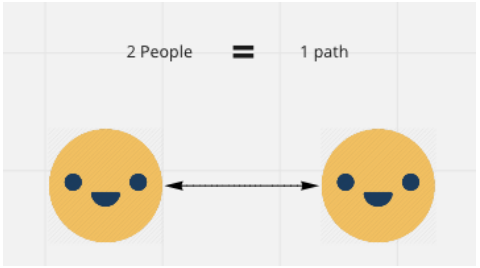
I _____ that communication is an inate characteristic as opposed to a possibly learned skill

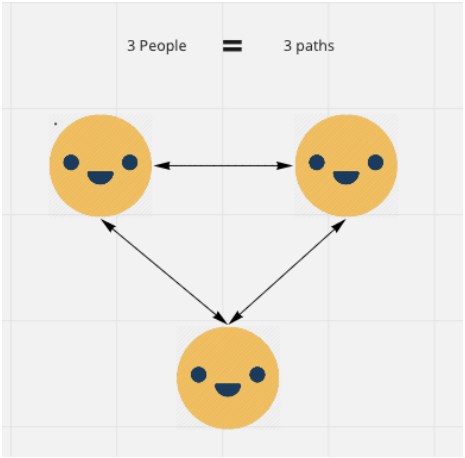


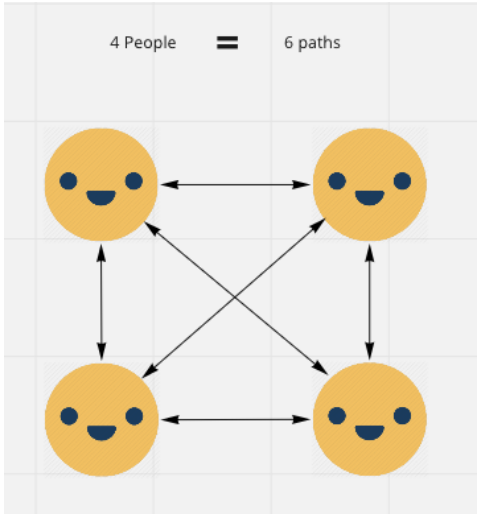
You Gotta Pass

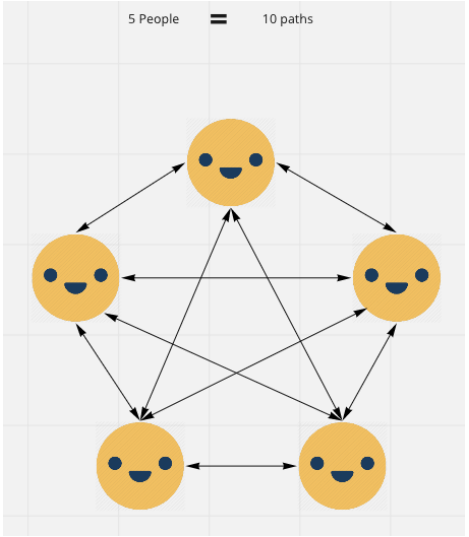
- Since I love soccer lets assume the idea of passing for a software team is communicating or working with each other
- In general we can't play alone but we also can't play "bunch ball".
- We need to learn how to pass, but this gets harder the more players we add

The More Players the More Paths

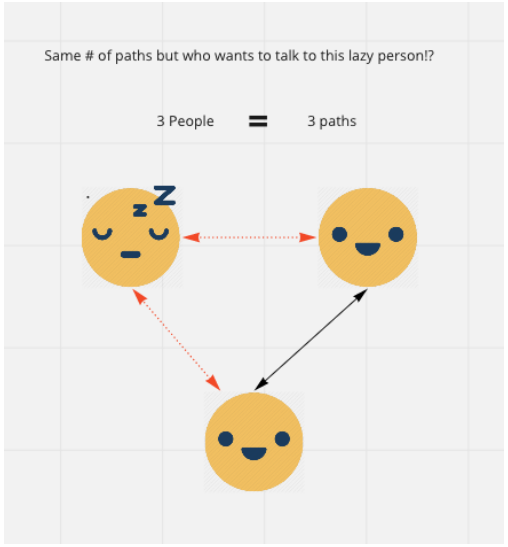


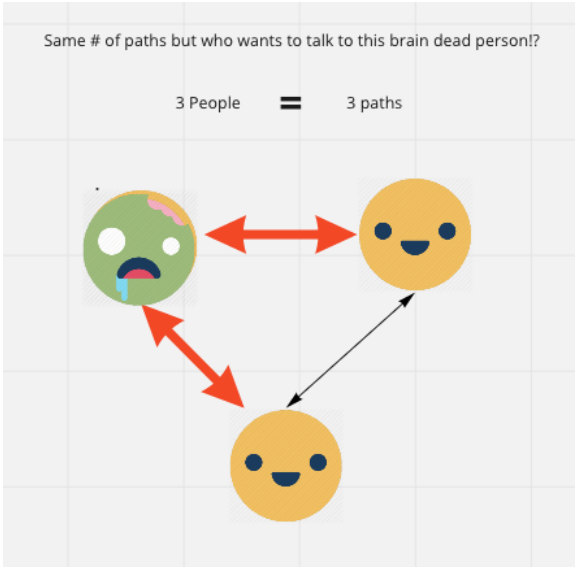




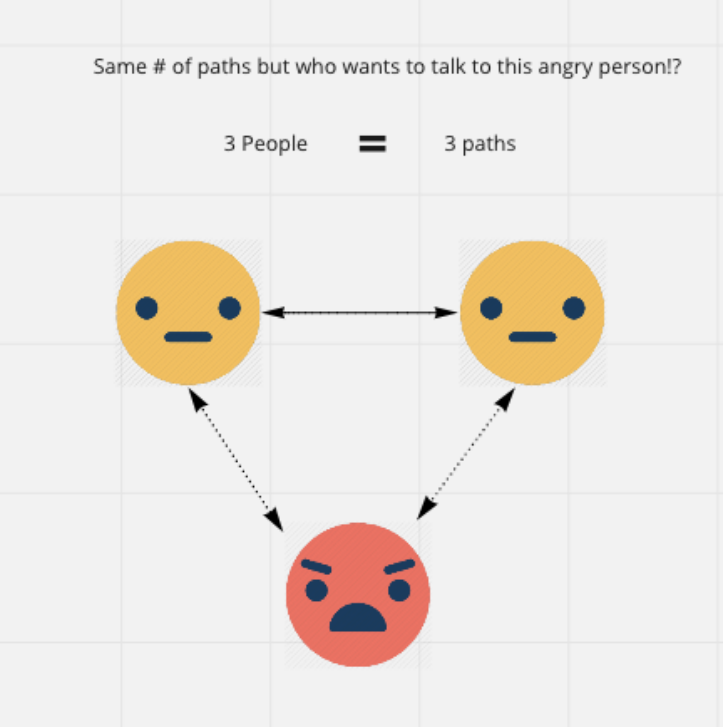


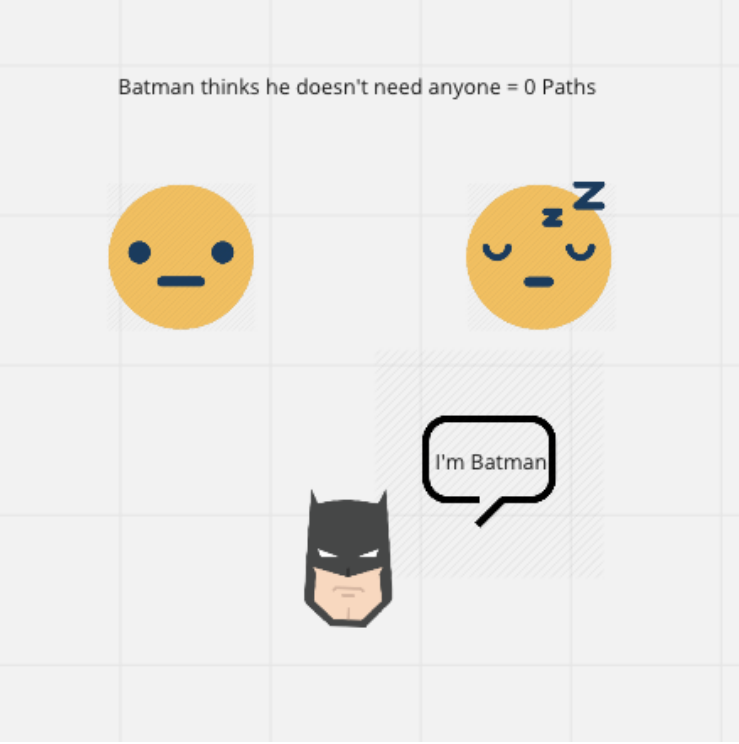
But this isn't Math



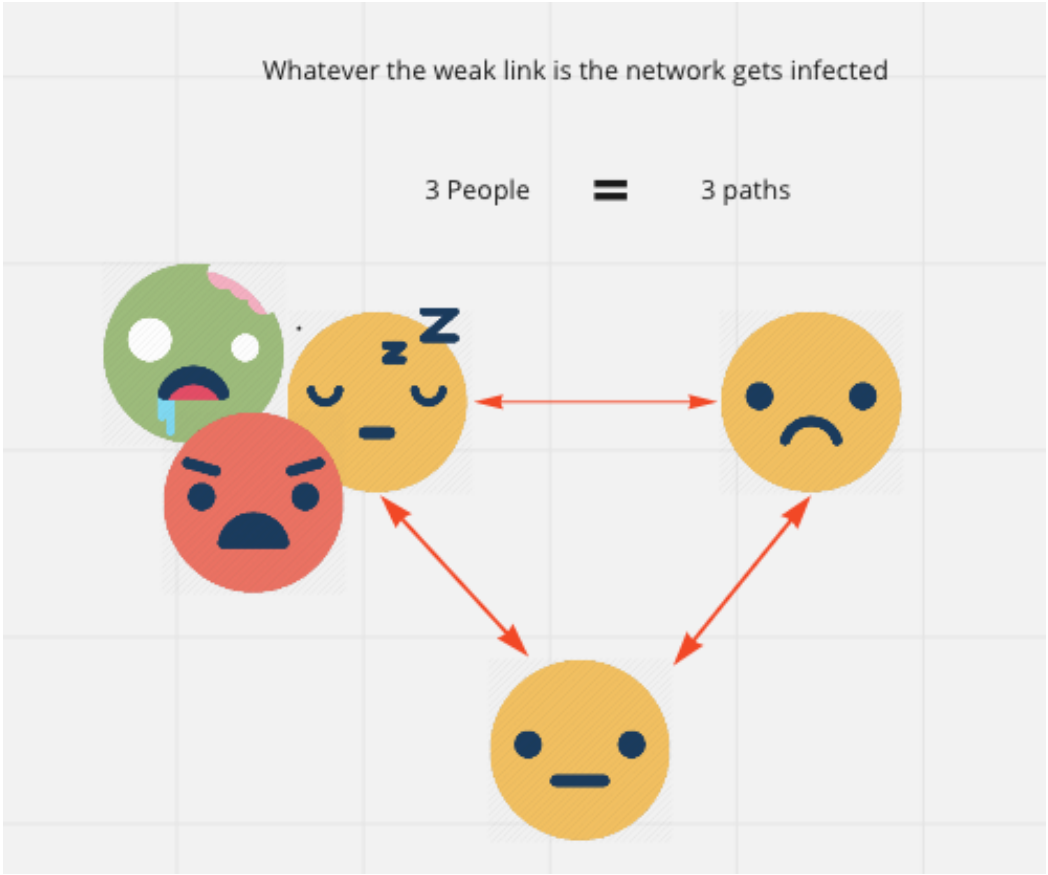


Yep still isn't Math

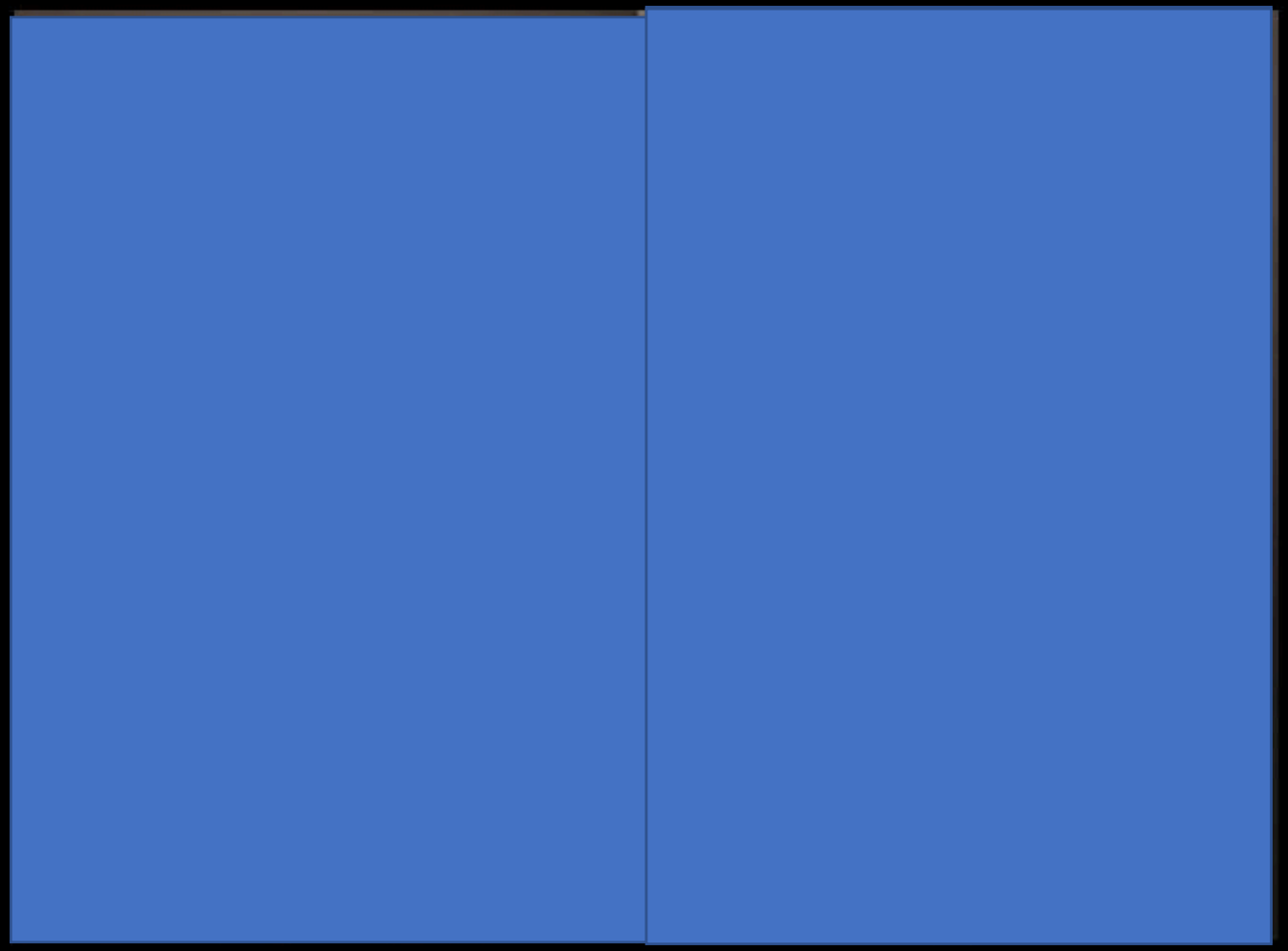




Infected!


Whatever the weak link is the network gets infected
3 People
=
3 paths

TRUST is Built on

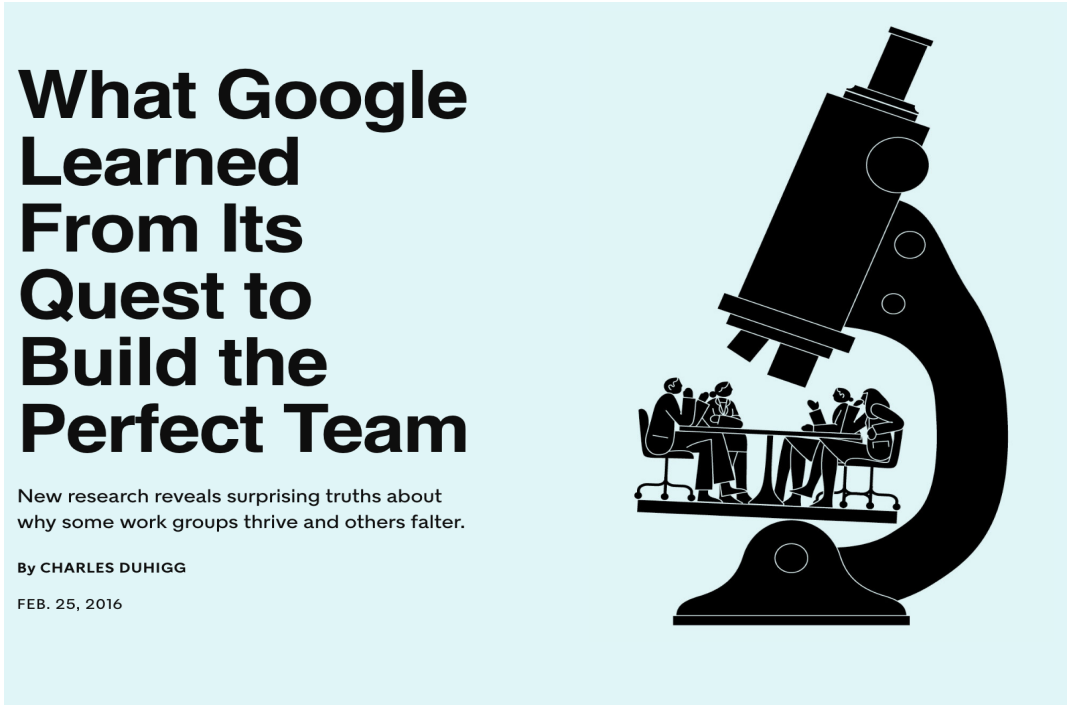


FIRST IMPRESSIONS

Beware of cliques

- Be careful with teams within teams – aka cliques
- Teams in our class with many previous experienced people can form cliques easily
- When a team works together very often it is quite possible it becomes clique-ish so that new team members can't get involved deeply
- Aside: The continued challenge of 'brogrammer' and 'gender problems' in our industry have been significantly amplified by such clubhouse/clique cultures. Watch out for "fit" being used as exclusion
- However, close working relationships that become this way are also an advantage!?
- So serious question – can we figure out teams? People think so, I'll say we can get some good clues but there are many ways to get to a "good team"

Data Driven Thoughts on Teams



- https://www.nytimes.com/2016/02/28/magazine/what-google-learned-from-its-quest-to-build-the-perfect-team.html?_r=0

The importance of things

The quality of the coach

The skill of the players

The hard work and practice put in

The morale or "teamnyness" of the team

How much they get or gain from winning (aka the Prize!)

The mindset of the group (aka their attitude)

The kind of gear/equipment they use

Their fan base

Results: 5 Things For Great Teams

1. Psychological Safety
2. Dependability
3. Structure and Clarity
4. Meaning of Work
5. Impact of Work

BUT... #1 apparently is paramount for the other 4

Psychological Safety

- Does the team have comfort taking risks?
  - Worry, shame, insecurity, embarrassment, etc. are going to really keep you from doing this
- Without this safety people don’t own up to mistakes and so problems fester
  - Same worries as above, admitting weakness is a “risk"

Building Psych Safety

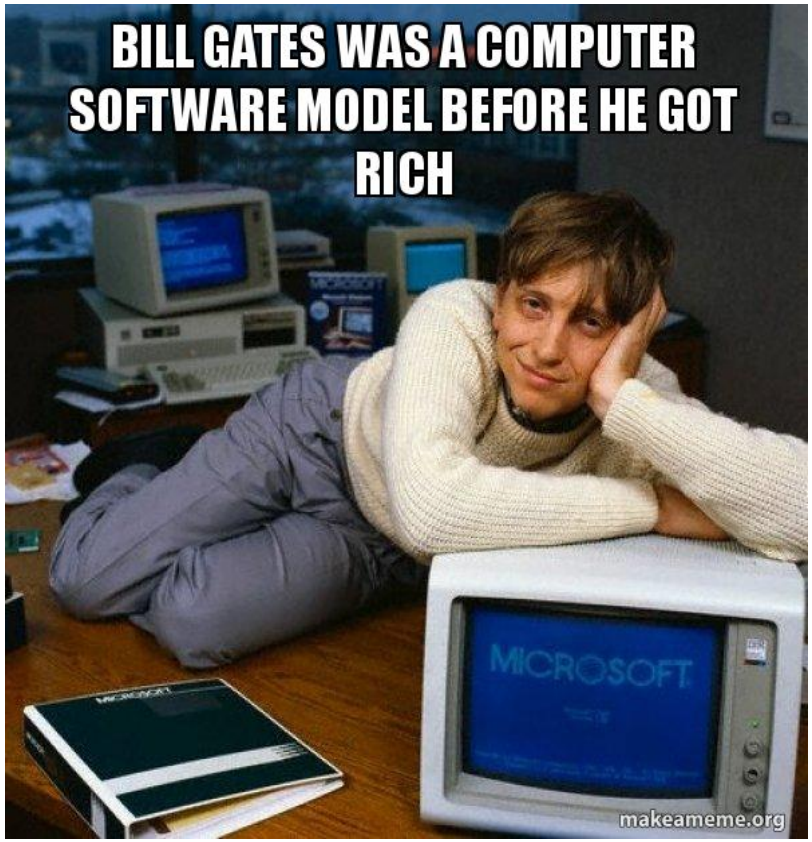
- Four general parts

1. Model
2. Allow Failure
3. Avoid Blame
4. Empathize

- It will take time and comfort obviously thus our team building efforts, various onboarding bootcamps, emphasis on “culture” at various companies, mixers, etc.

Modelling

- Try to show and do as opposed to tell
  - Leader: “I command everyone to be nice to each other!”
    - Yeah that won’t work
  - “Everybody is being such a jerk right now!”
    - Yeah you too
  - Instead…
- Be a role model yourself
- Leaders can have biggest impact, but all team members can do it
- Be humble, vulnerable, willing to make mistakes, etc. and set the tone



Failure is an option

- You must remove the fear of failure
- Change goal to a focus on learning and experience
- Of course when you do be prepared fails will happen!


All aboard the Failboat


I DON'T ALWAYS BREAK THE BUILD
BUT WHEN I DO, I BLAME IT ON YOU.

Get Off the Blame Train!

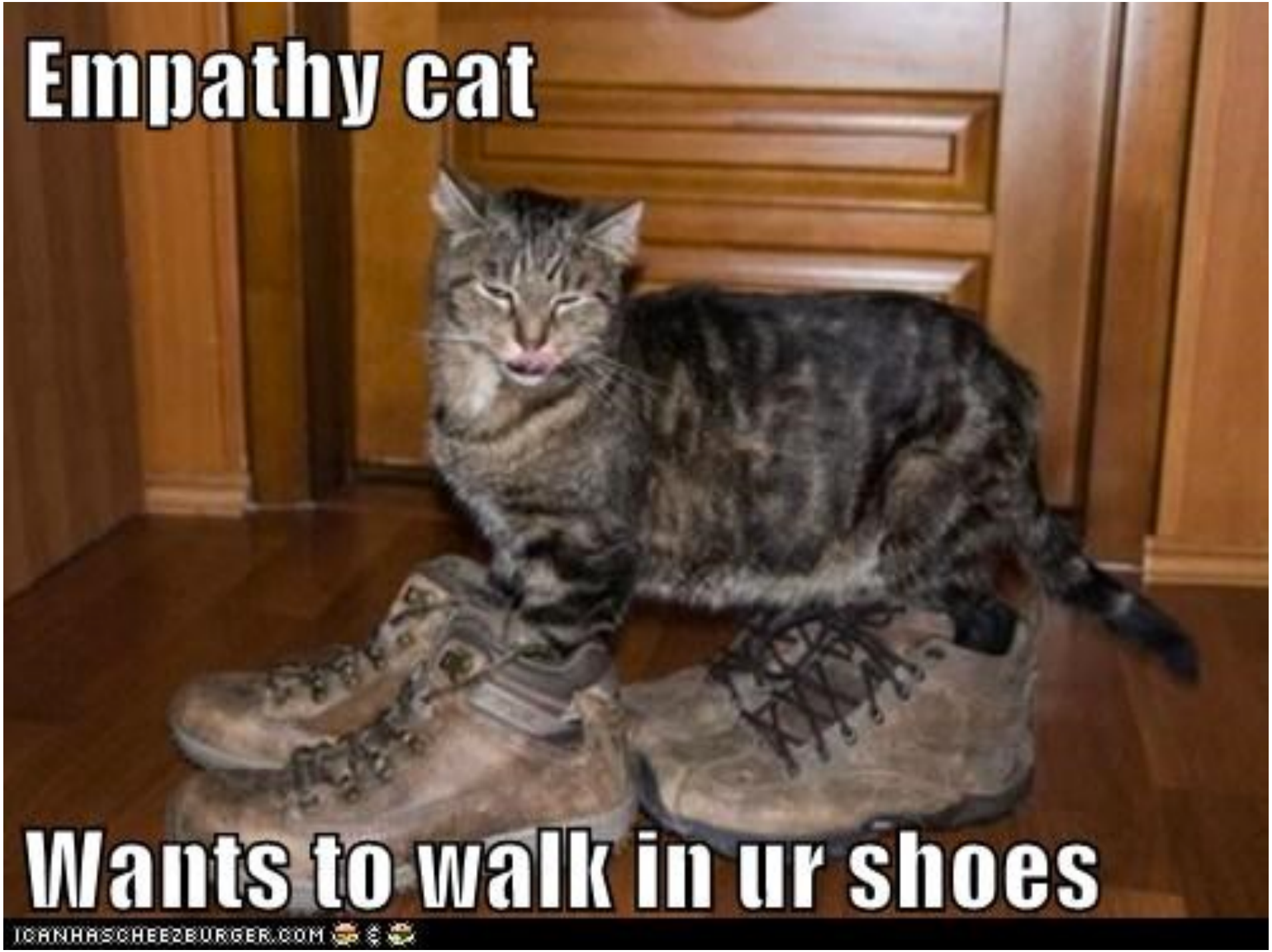
- Stuff happens get over it!
- Remember a ‘pass’ given to a teammate might be redeemed in the future for your mistake if you don’t blame them now
- If you allow the blame to happen too much, then people start to fear revealing failure and psych safety starts to go away
- We really need to have some compassion & empathy for our teammates

EMPATHY

would this help?

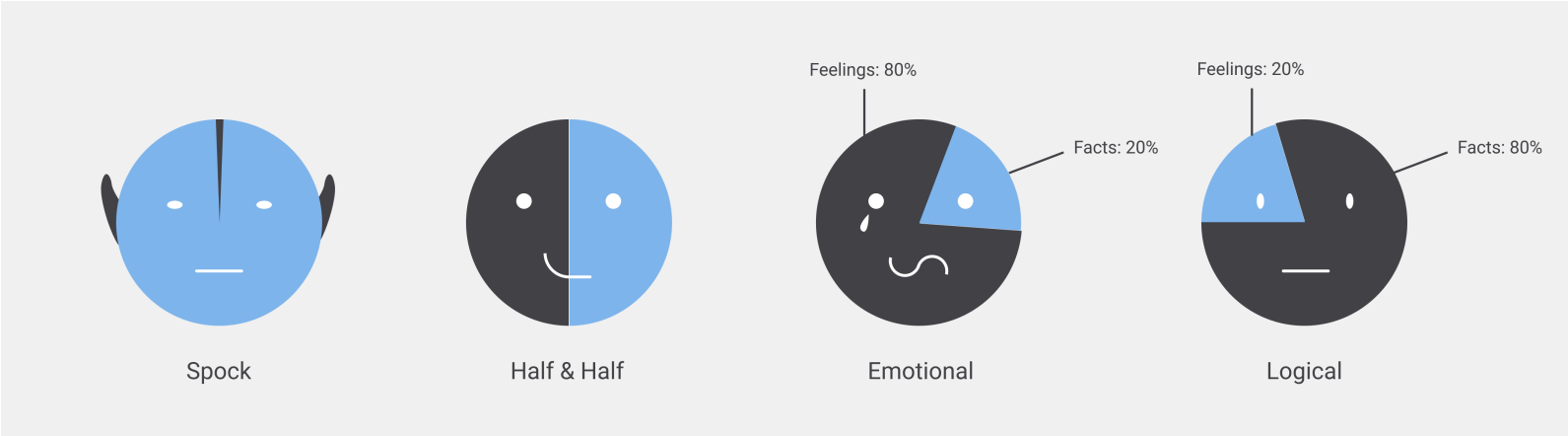
Empathize

- Everybody is different
- Try to ”walk in their shoes”
- Remember nobody* is trying to be stupid, evil, etc.
  - Yes there are certain toxic (or even sick/criminal) people but we are not going to let an exception spoil the norm
  - Beware without empathy toxicity creeps in and with that teams can ‘die’


Empathy cat
Wants to walk in ur shoes
ICANHASCHEEZBURGER.COM

The “F” Word Varies for Us

F being for feelings!



Each of our teammates may have different personal “pie charts”

The other 4 points

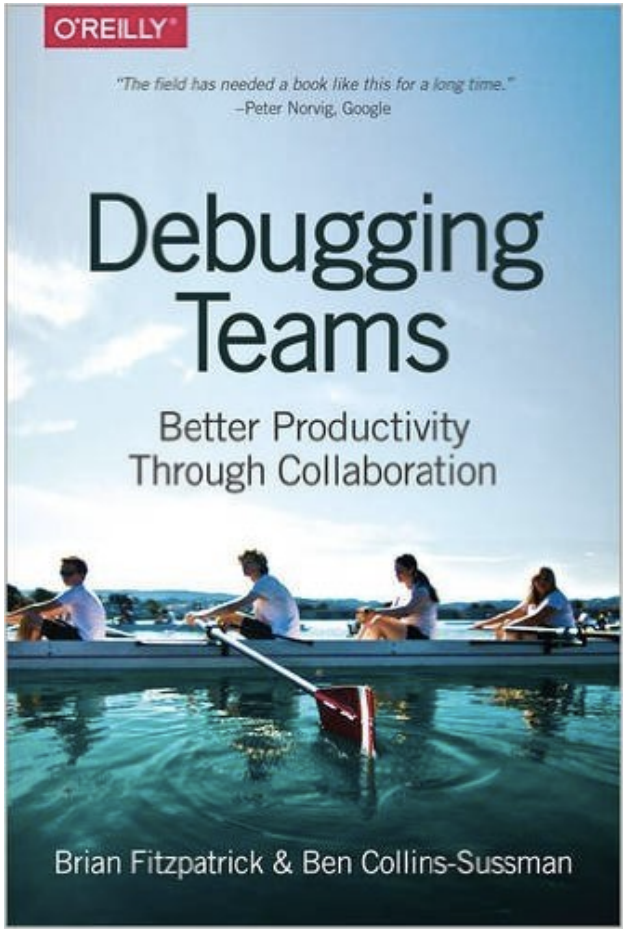
- Dependability
  - Remember the skip the meetings problem!
- Structure and Clarity?
  - What happens if we don't have that from life?
  - Who then gives structure and clarity?
- Meaning and Impact of work
  - "But I'm just doing the _____" (docs, testing, unit tests, refactoring, etc.)
  - Kind of a bad attitude
    - "Coach I don't want to play on the team unless I'm the striker!"
  - Shouldn't you be like the NASA janitor?
    - JFK to Janitor asking about what he does
    - Janitor: "Well, Mr. President," the janitor responded, "I'm helping put a man on the moon."
  - Not every project or app is a game changer (or is it to someone at least?)

Smells of Teamy-ness

- Difficult to pin down like what is a “friend”, we know when it is or isn’t but there are signs. Google’s data did point out a few things and I added a few of my own *
- A Group Intelligence and Ownership emerges
- Use of “we” over “I” *
- Small member domination is lesser (more even number of communication points)
- People don’t tend to talk over or ”at”* each other
- There are shared views
- Interestingly Google seemed to find that have a better gender density correlated with team success in their study as well.

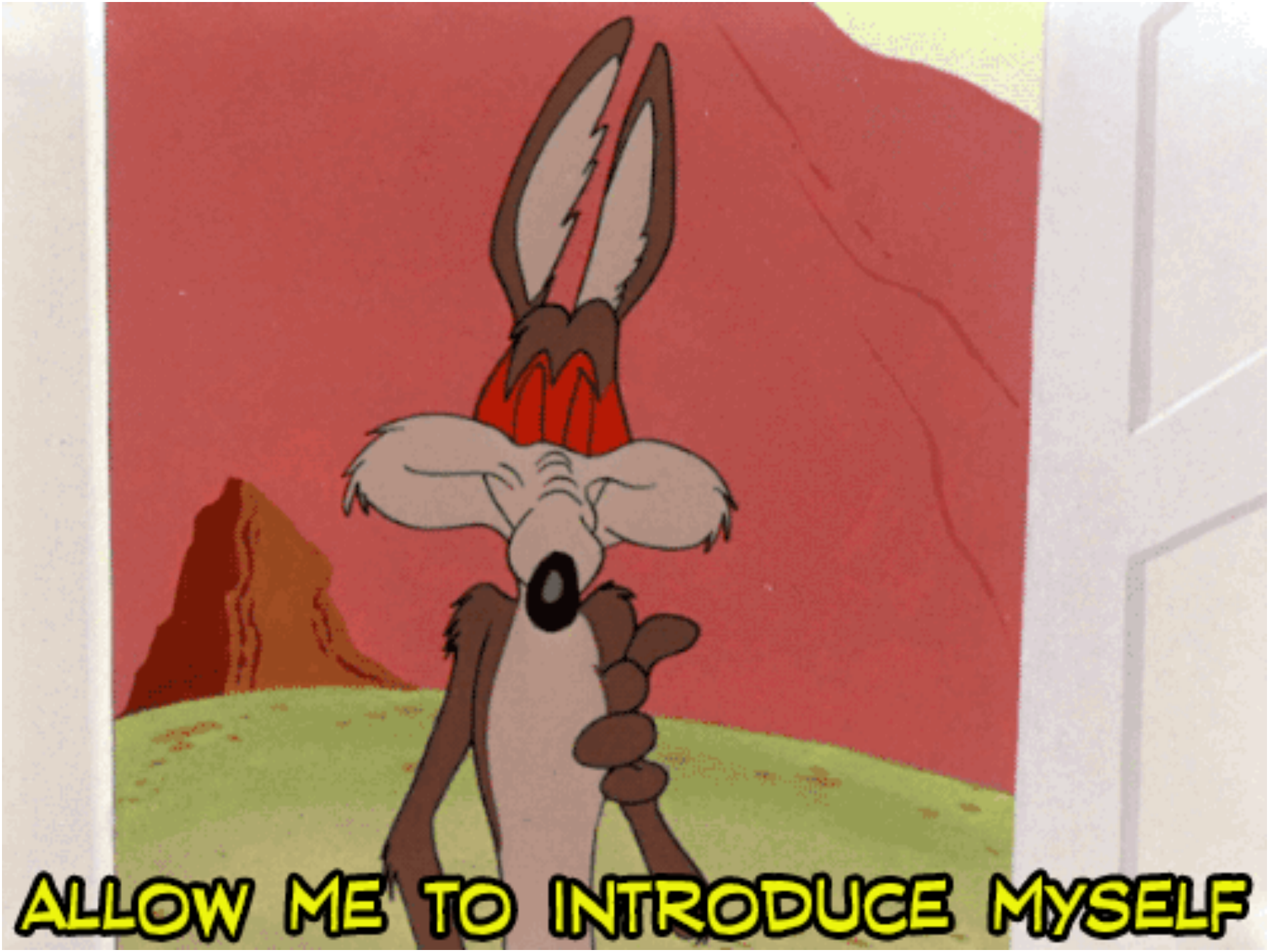
Debugging Teams

- If you believe some of the premises from last lecture the secret weapon a software engineer needs to master is often softer than code!
  - Much of the content out there focuses on programmer self-improvement. Following Postel's Law for yourself that makes sense!
  - Today's content tries to address team issues, though it starts of course with you again
- These slides represent some of the findings from *Debugging Teams* by Brian Fitzpatrick & Ben Collins-Sussman with numerous items from other books like *Joel on Software* and the Prof's experience mixed in for good measure



Geniuses

- A lot of our problem in groups starts with ourselves as we saw last time. Let's start with the rough root cause of many problems for programmers
- Subscribing to the Myth of the Genius Programmer
- Makes sense in a perverse way if we really equate our self-worth dominantly with what we code
- Unfortunately, the emphasis on self is a myth as even recognized geniuses are nothing without a team
  - Jordan, Jobs, Gates, Torvalds, Musk, etc. without their team(s) they are not nearly as powerful as with them


ALLOW ME TO INTRODUCE MYSELF

Hiding

- When we subscribe to this apparently noble intent bad things can happen
  - From last lecture this leads to the impostor syndrome in extreme
  - Less extreme leads to code hiding and lack of communication
  - Ex: “I don’t want people to see what is clearly a work in progress, otherwise they might judge me!”
  - Ex: “I am not sure my idea is any good, so I’ll just say nothing”
- Hiding also leads eventually leads us to information hoarding, which then build us into fake ‘geniuses’ and may lands us job security by information obscurity.
- Our intentions while self protecting end up backfiring on us

Bus Factor

- Bus Factor – A numeric value that represents the number of people on the team/project that if hit by a bus would doom the project
- A low Bus Factor represents extreme project risk!
- Bus Factors often grow with info hiding and hoarding
- Sometimes teams accidentally worsen their bus factor
  - "Jennifer is the best at the build scripts, we will just let her work on it since that is most efficient"
  - "It doesn't make sense for everybody to do everything, so we will each specialize ourselves"
- This is bad Engineering at the team level – there is no redundancy or fail safe at the human level! Sports, symphonies have back-ups and subs does your team?

Bus Factor Story

Senior Engineer X: I really think I can do better than NodeJS, it will be your PHP in JS idea. Let's do this and build something amazing.

Repeat for 2 years with demos

Me: (after finally deciding to go for it) Ok, let's do it

Senior Engineer X: Awesome! About time, let's do this.

6 months pass, code is created with a bus factor near 1 and baked into systems and then…

Senior Engineer X: I am quitting. I'm bored and this isn't going anywhere.

- Result: Project was doomed and looked to never really work in a deep way without significant time/$ investment. To recover 1 -2 developer year+ effort to unroll in-house code and revert to NodeJS

Programmer Life Cycle?

New Hires start at step 1

1. Ambitious
2. Productive
3. Irreplaceable → BUS FACTOR RISK!
4. Resentful → QUIT
5. Bored → QUIT
6. Unproductive → FIRED

Is this a microcosm of how some group projects go?

Let's avoid getting past step 2 or so and if we do it makes sense to "bail before we fail"


THIS PLACE WILL FAIL ONCE I'M GONE!
SAID THE GUY WHO NEVER DOCUMENTED ANYTHING
imgflip.com

You Have to Work With Others

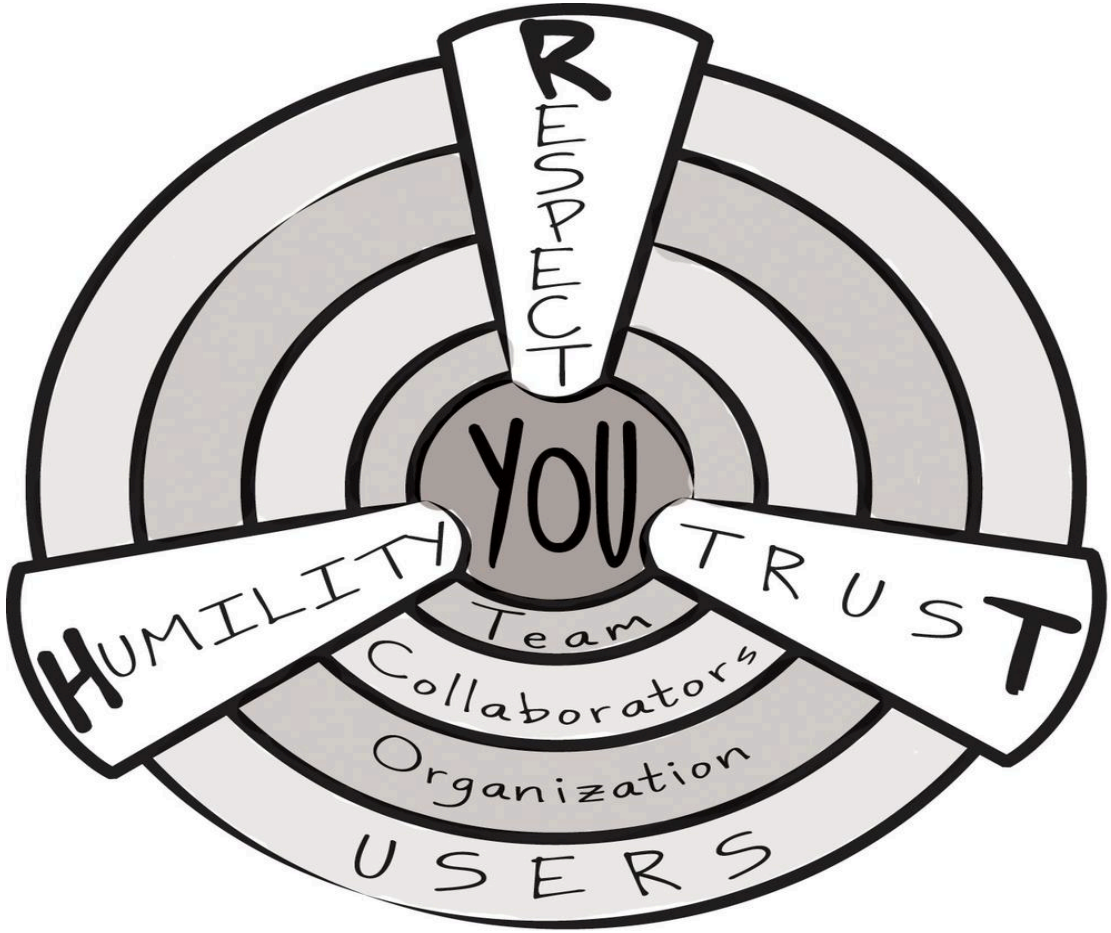
- Any sane organization shouldn't let you become the invaluable hoodie wearing hermit developer
  - Some will but they shouldn't as things can and do go bad
- You can do it on your own clearly, but you shouldn't
- You learn from others which makes you become a better programmer
- You can get more accomplished with a team than by yourself

Working alone is just riskier than working with others!

HRT (HEART) Method

- **Humility** – you aren't perfect, you should be open to self-improvement and growth
- **Respect** – you appreciate others, their intentions and their efforts. You treat them as fellow human beings with feelings just like you
- **Trust** – You trust that others are competent ad will do the right thing if you let them

HRT Visualized



HRT doesn't solve, it explains

- Social conflicts generally can be traced back to a combination of these three pillars
- Understand that since there are many layers here it requires that everyone acknowledge these pillars otherwise it doesn't do much
  - Translation: You can employ this, but others may not, if they don't, they should be ejected from the team, collaborators group, organization, or user population

Tips on Living HRT

- Lose the ego
- Giving and receiving ~~criticism~~ feedback
- You aren't your code
- Fail fast and iterate
  - Not just for code but for social interactions
- Think Binary
  - Change the scale and make it a 0 and 1 outcome
  - Something simple, did you do it yes or no not something large that leaves room for failure
    - Doorknob turning ("yeah, it opened!") vs a self-critical discussion of doorknob turning ("my wrist wasn't at a 37degree angle and it took more than 1second and plus I wasn't really happy about it")

More Tips

- Leave time for learning
- Be patient
- Learn to listen
- Be open to influence
  - Changing your mind isn't weak ,it's mature
  - Say I don't know

- Think Soccer, ~~Dempsey~~ Morris may have scored the goal, but the Sounders won not just ~~Dempsey~~ Morris.
  - The team plays not just the "star" player (you or otherwise)

Meetings

- General Tips
  - Only invite people who **need** to be there
  - Have an agenda and share it before the meeting start
  - Be time aware, start and end the meeting on time, if done early, end early
  - Don't let the meeting wander (too much – allow outlets)
  - Back meetings against interrupts or unproductive time slots
- The stand-up meeting form – StatusHero Discussion
  - https://www.hugo.team/ and Notion.so, etc. (see slides upcoming)
- In-person vs. Online
  - Non-verbal communication
  - Example: Remote only vs Local and then remote
- Maker's Schedule vs Manager's Schedule
  - Managers – interrupt for status
  - Makers - 4-hour blocks or more!
  - Headphones as your office door?

Programmer Communication

- Synchronous vs Asynchronous
  - Online allows for asynchronous, real world not so much, tends to be rude
- Meetings
  - In-Person
  - Online – Slack (yeah it has it), Zoom.us, Google Hangout, Skype, etc.
- Group Chat
- 1-1 Chat
- Email
- Email List and Discussion Thread
  - Noisy minority problems
- Design Doc
  - A living document, not stone tablets
- Issues
- Code Comments
- Reviewing Comments

A fundamental mistake

- Your product / program is ultimately about communication, not just with a machine
  - Machine – yes it needs code that is syntactically correct, but in some sense, it doesn't care how well developed your code is. It doesn't care, it's not alive. You and other humans care about runtime efficiency, optimization, maintainability, etc.
  - Developers – you and other people who read your code and work with you on your code
  - Stake Holders – people who guide the product and thus your code and likely are what funds your code
  - Collaborators (vendors, interested third parties, loose stake holders) – people how have some vested interest in your code
  - End users -

Software >>> Code

Powell's Power Set Theory for Software

- Software is the power set of many things including:
  - Source Code
  - Test Cases
  - Requirements and Plans
  - Issues
  - Documentation
    - Internal and External
  - Supporting Documents and Communication Assets

- A common failure is not addressing a single source of truth for other aspects of your code

Single Source(s) of Truth

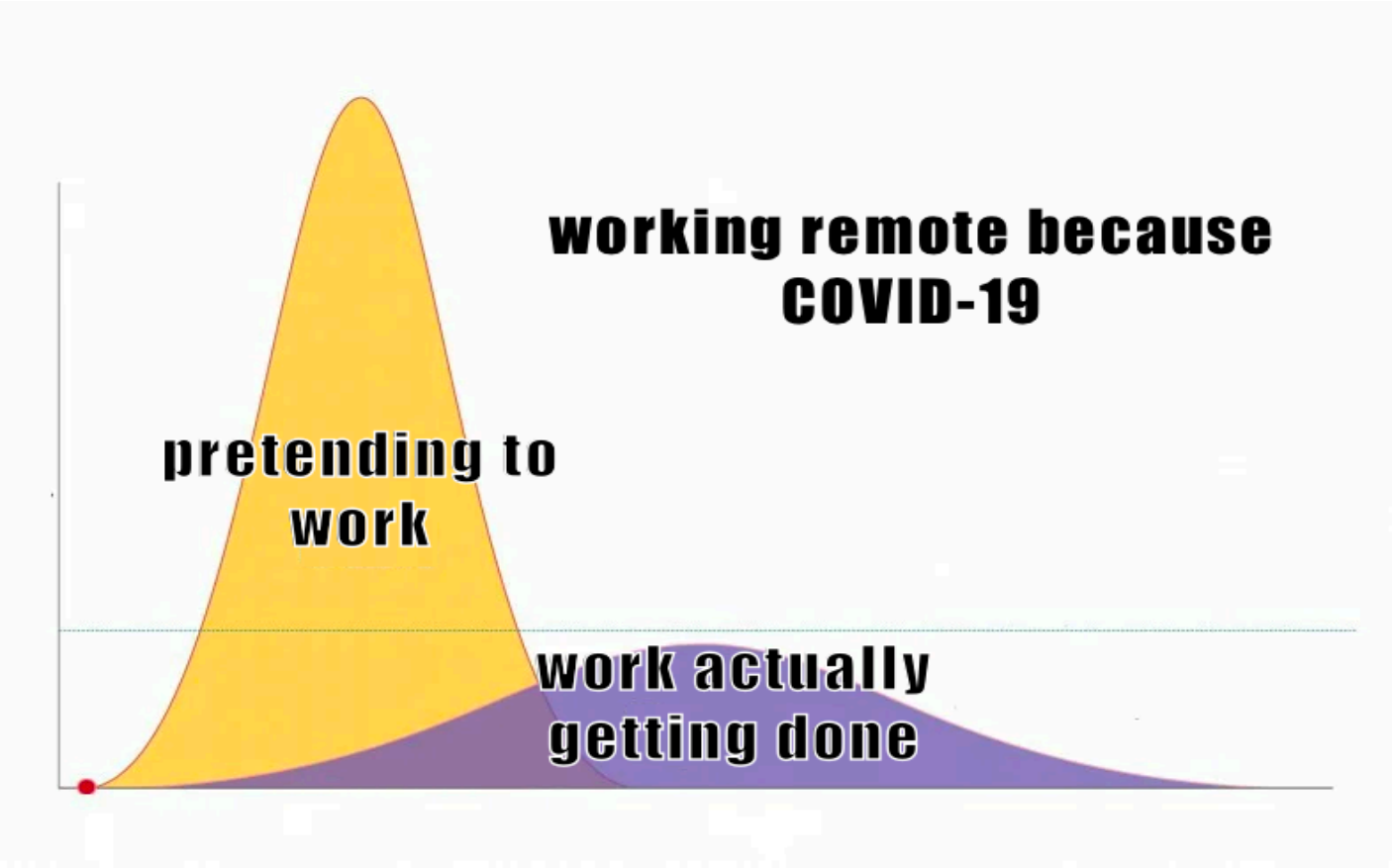
- The (s) suggest it is misnamed but I'll allow for it
- With code we subscribe strongly to a single source of truth – the **Git Repo!**

- Given the power set you need some single source concept fore more than code or is it a tree with a primary entry (ex. Intranet/Slack) and then "single sources" for the domain or tasks at hand

- For me this is still open and the tooling out there seems to favor a distributed idea
  - Fail? Multi-tooling leads to multitasking = waste

Project Preview!

Flatten the Curve?



Seriously…This is Really Hard!

teacher: does everyone understand???
everybody on zoom:

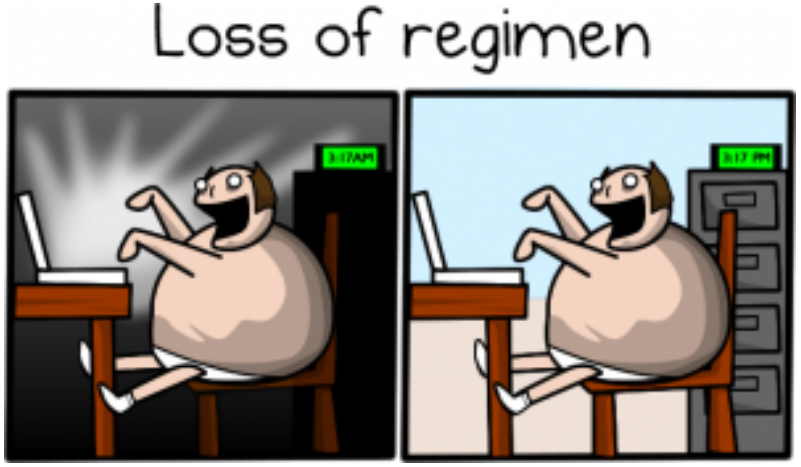
Probably the only silver lining is…

MEME CREATIVITY! IT'S OVER 9000!

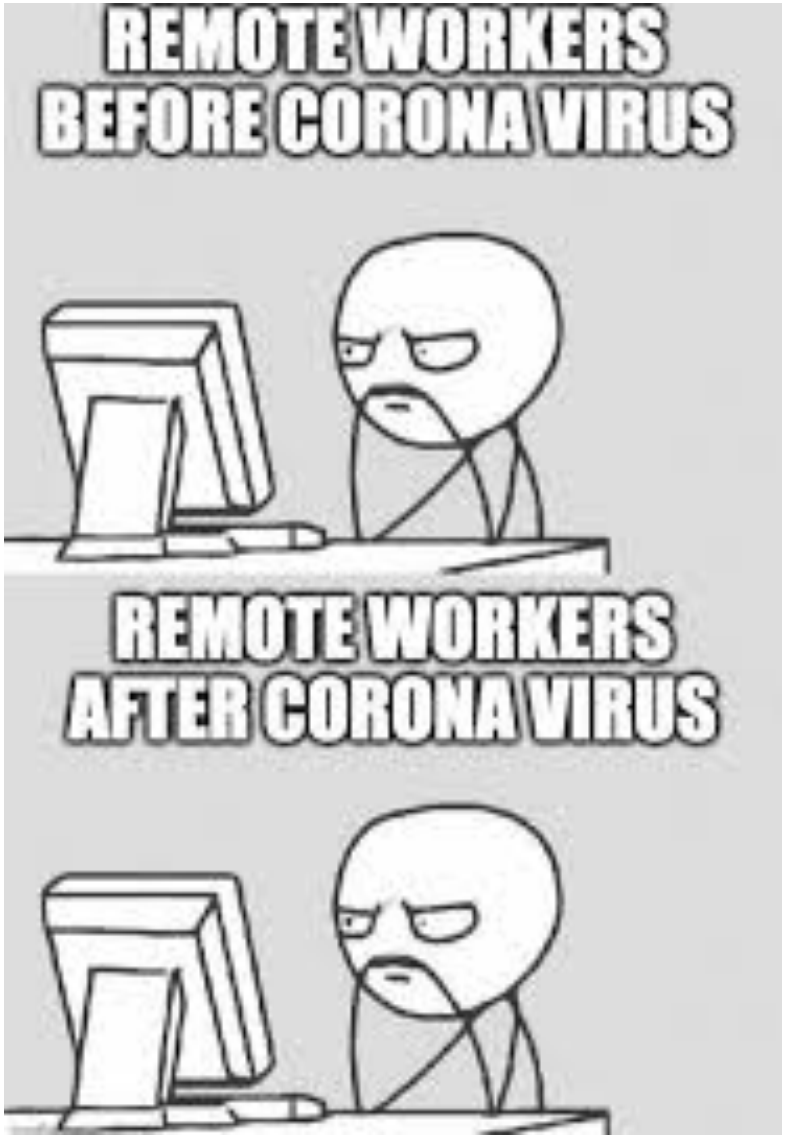
I know…

We have good intentions…

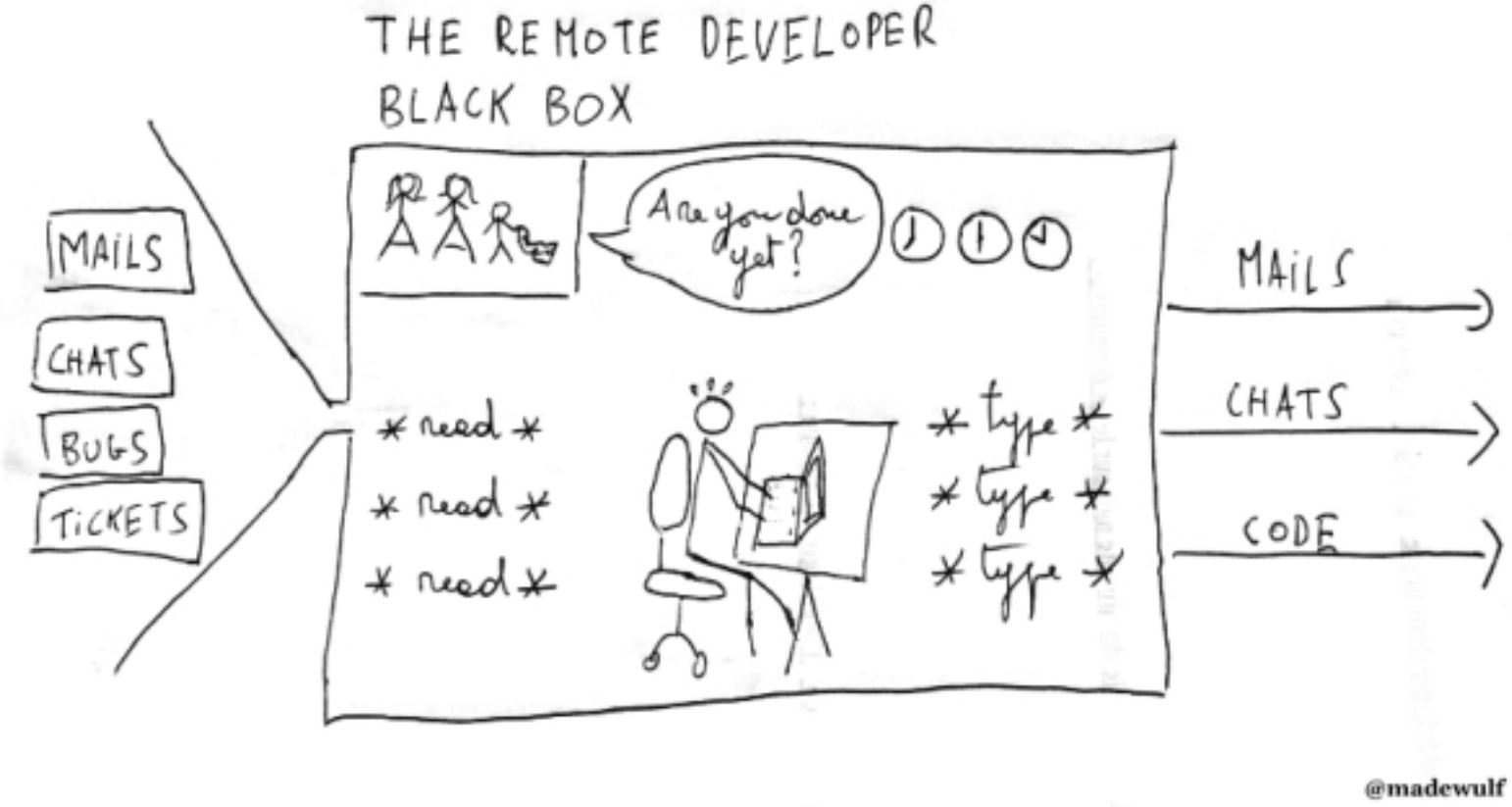
but admittedly lots of challenges…



On Groups – PollEv.com/TPOWELL

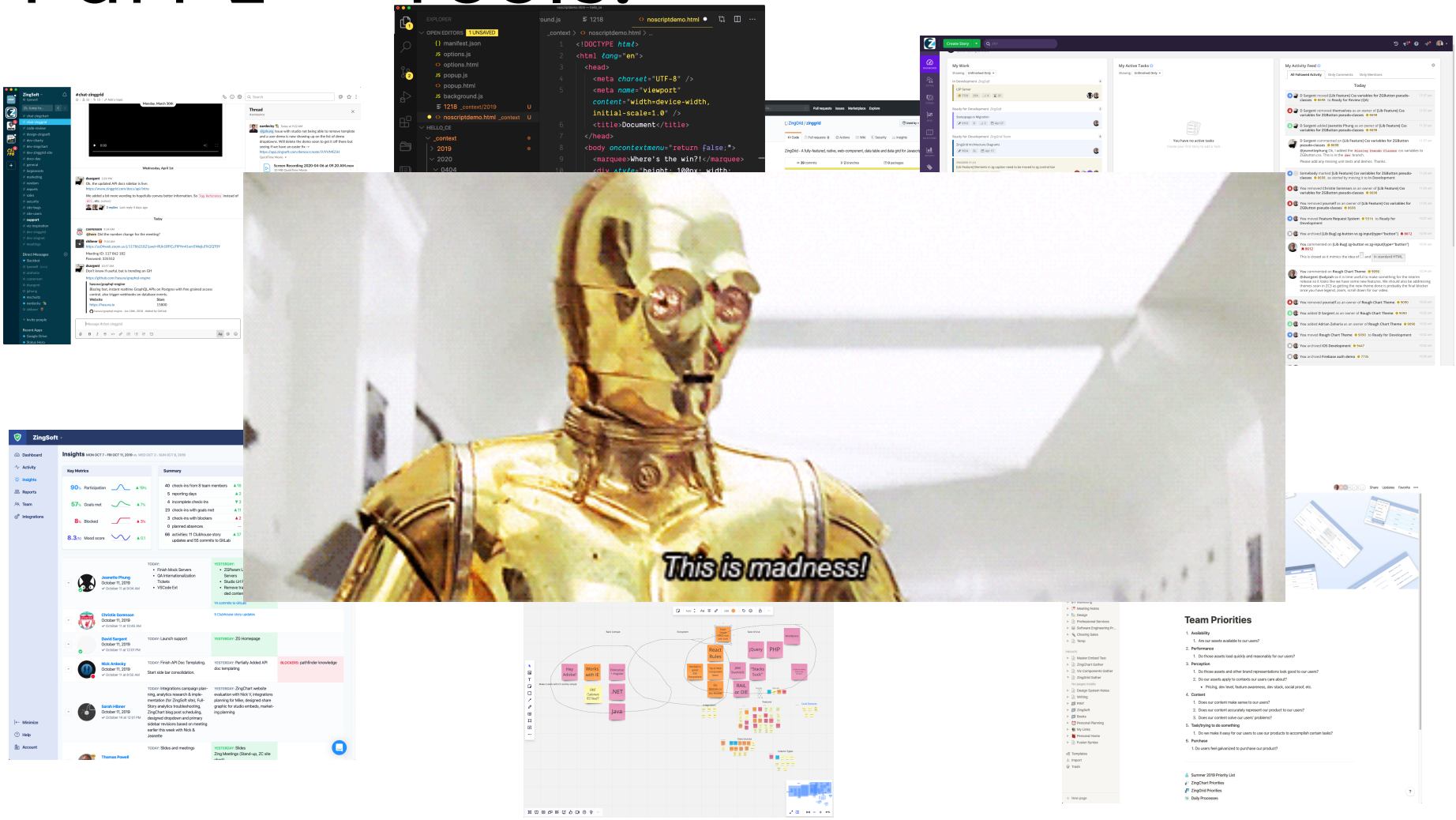


Pretty Accurately Visualization



Part 1 - ”Hey”

Part 2 – “Tools!”



Part 3: One True Thing!

- Code - Github
- Communications – Slack, Basecamp
- Plans & Docs - *Notion, Github Wiki, Google Docs/Drive, Others*
- Tests – Github?
- External Docs – Github / Site, Notion
- Management – Slack (?), Dashboard (?)

- For me, a big tip is understanding that this is where I have tended to go wrong and and if I could fix things it would be some of this

Let me know very briefly your initial impression of these problems (and maybe solutions) next Tuesday or Thursday during the intro!

More will come later...

Back to the deck...

Don’t Focus on the Particular Game So Much

- Let’s do some liminal thinking
- Looking at other things you will see some of the absurdity we have about making effective teams
- Let’s use soccer since I have a few times, but if it doesn’t resonate with you pick some other activity that is with a group of people

If a soccer team lost a ton of games over a season

The coach should be fired

The players should be fired

Everyone should be fired

We should try to understand who to fire

Nobody should be fired

My feelings about coaches effecting soccer (or other sports) teams is that ....

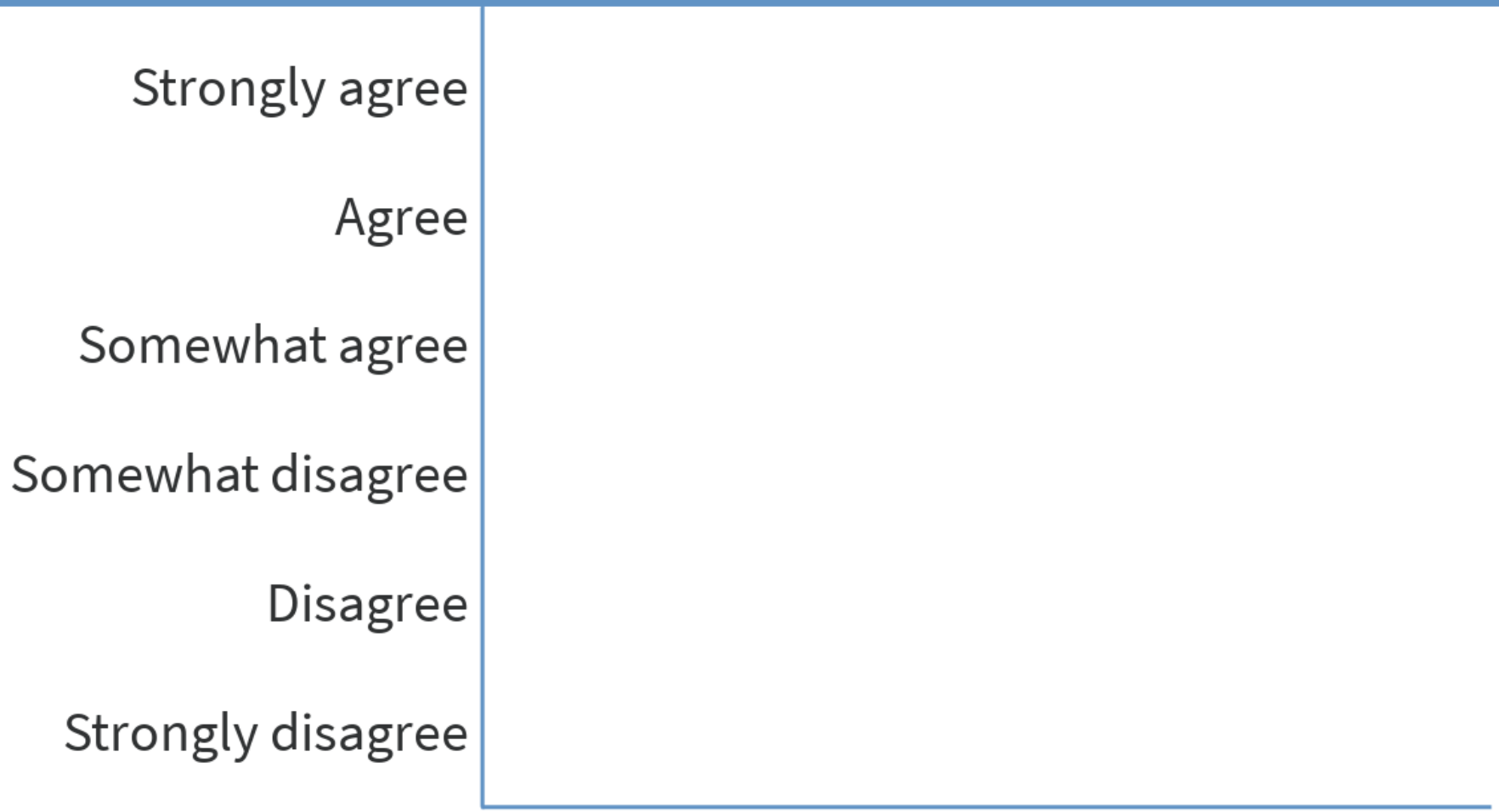
Coaches can strongly effect the team outcome

Coaches can somewhat effect the team outcome

Coaches only slightly effect the team outcome

Coaches have little to no effect on team outcome (It's mostly the players)

In order to be a good coach at something you have to be (or have been) excellent at that thing (use soccer, basketball, etc. if you like)



I feel a manager is like a coach

Yeah they are kind of like that

Maybe there are certainly some overlaps

No way this is coding not sports or other activities it is different

We do need managers

- Like it or not you need managers
- A group of people can do more than a single person in most cases unless extreme disfunction
  - Simple labor output
  - Specialized skill (generalists vs specialists)
  - Full Stack Developers vs Full Stack Aware Developers
- Teams often underperform though because problems of coordination and motivation chip away at collaboration
- Managers (when doing their job) can get better outcomes from a group of people working together

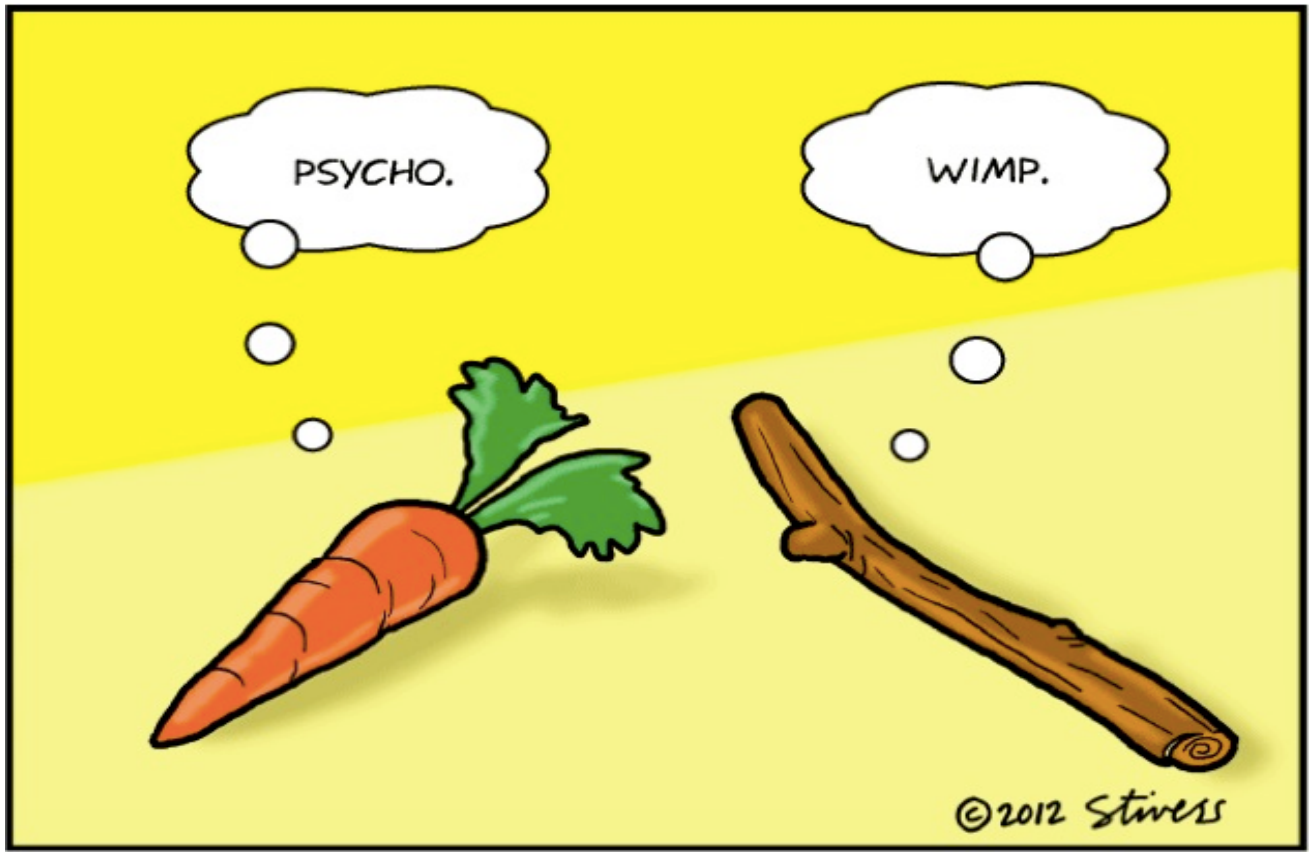
Words and ideas about bad managers

Words and ideas about good managers

~~Manager~~ Leader

- In short, we really do need them, without someone in charge we drift around pulled by random currents
  - Reality: A leader will generally happen either by appointment or self-selection
- Even if you never plan on being a manager/leader which is quite fine you would be wise to understand them
  - Helps you understand their decision-making process
  - Leads to better communication and often better reviews (think raises)
- Good idea to understand motivation https://www.youtube.com/watch?v=u6XAPnuFjJc
  - This might defy traditional motivation techniques (next slide)

Stick and Carrot



(If You Need to) be a Manager

- Being a leader is tough. It really isn't for everyone, but like all things it will take practice
  - Great opportunity to do so with safety here in school!
- Code is tangible, there is a result and so inherent makers will find it satisfying
- Management lacks quick results and is difficult to measure and can be less obviously satisfying
  - Realization: Less obvious growth except over time, have to know the signs to look for
- Being a manager allows you to scale yourself in some sense, help bring your version out maybe only way to do it
  - You would like humanity to get to Mars? Seems cool, maybe you think Elon might be able to do this, no strike that thought Elon might get his team to help him do it!
  - What kind of manager should we be?

Five Conditions to Foster

- Five conditions are likely helpful to improve the team's odds of victory

1. Clear team with boundaries and stable membership
2. Compelling and understood direction
3. Enabling structure
4. Supporting context
5. Expert coaching

Three Areas

- An easy what to think of what is necessary in the team is WWH
  - **W**hy
  - **W**ho
  - **H**ow
- More practically – Apply 3Ps
  - **P**urpose – why are we doing this
  - **P**eople – who are the people that are doing this
  - **P**rocess – how are we going about doing this
- Process should come last.

Respect the Three Time Zones

- We need to live in all three time zones – Past, Present and Future
  - Too much in past leads to …
  - To much in future leads to …
  - Just living in the present leads to …
- Learning from the past helps keep us from redoing mistakes and guides better decisions
- Planning for the future helps us plot a course and motivate our team in a clear direction
- Living in the present allows us to be grounded to what is important right now

Manager Math

- Managers should aim not to be just additive
  - The fallacy of the working manager
- Manager should really aim to be multiplicative
  - A good manager magnifies the output of others which is vastly more important than the incremental value they might add them selves additively
- Bad managers might be subtractive
  - Get in the way of the team and reduce the output
- Really bad managers might be divisive
  - Destroy the team with dividing blame

Is Management for you?

- Avoid doing it because it is a “promotion”

- Be a manager because
  - You are at least ok if not like talking to people
  - You are ok trusting and promoting others
  - You think you are stable and calm

- Also
  - You think you have something to share
  - You want to try something different
  - You are up for a challenge

Leadership

- Leadership is a quality not a job
- Managers need to have it, but in reality, all members of a good team should have leadership characteristics
- Leadership is something earned not given
- All students in this class should aim to be a leader in some sense, at the very least in whatever part of the team they end up working on
  - Leader of code, leader of docs, leader of test, etc.
  - Motivation of cross team peering hopefully now understood?

BRAND CAMP

by Tom Fishburne

THE 8 TYPES OF MANAGERS

NOW THAT I'VE MADE NOISE AND DUMPED ON EVERYONE. I'M OFF AGAIN

THE SEAGULL

I PREFER TIMES NEW ROMAN

THE MICRO

CAN YOU MAKE SLIDES FOR MY AWARD SPEECH ON YOUR PROJECT?

THE SHOW BOAT

USE ONLY THESE TALKING POINTS WITH MY BOSS

THE SPIN DOCTOR

THE ABSENTEE

HOPE YOU DIDN'T HAVE WEEKEND PLANS

THE PITCHFORK

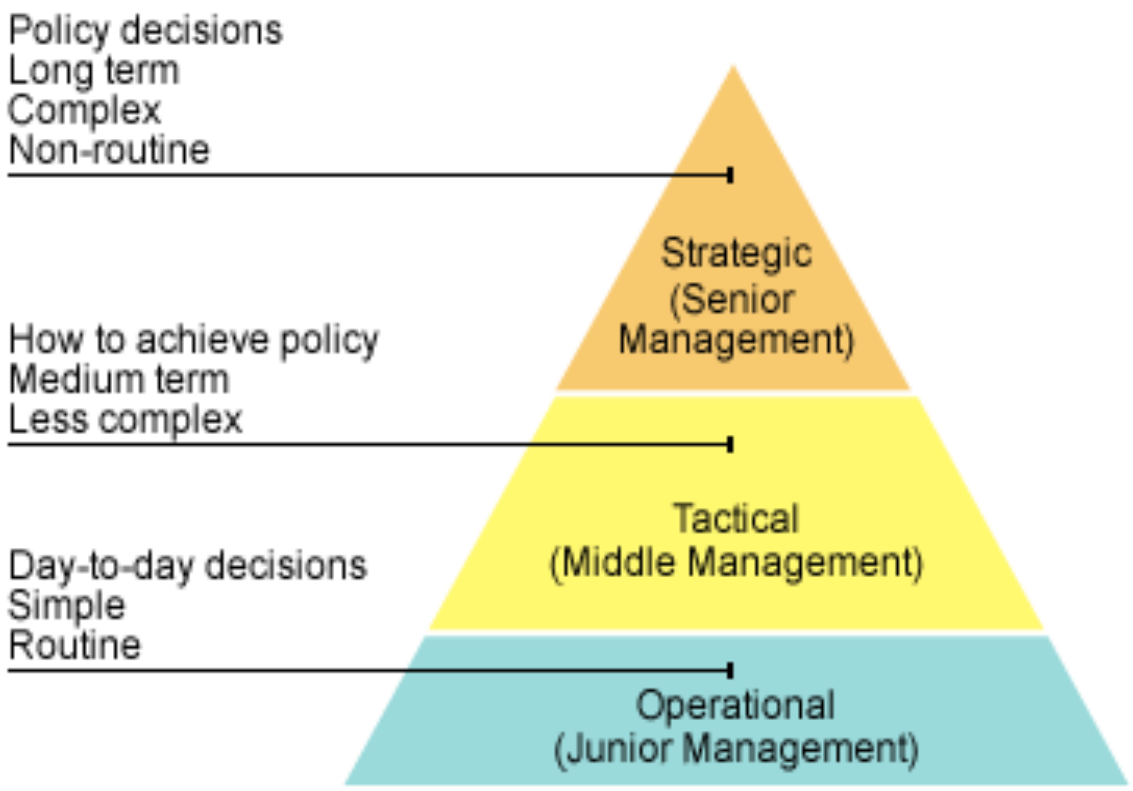
BEFORE WE DECIDE THE OFFSITE THEME, LET'S REVIEW THE GARTNER HYPE CYCLE

THE PONTIFICATOR

I MADE CUPCAKES TO GO THROUGH YOUR REVIEW

THE PLEASER

Management Changes by Level



Tips on Being a Decent Manager

- First - adopt the right attitude to be a great leader
  - servant leadership vs management
- Second – don't lose sight of your managed craft
  - Leading from the front so to speak or bags on contractor, coding manager, etc.
- Third – lead from the front
  - Act don't just tell, always be there for the team (see #1)
- More tips on trying to be a reasonable leader
  - Lower your ego
  - Be calm - change is constant anyway, don't freak out!
  - Ask lots of questions

More Tips

- Listen
  - My master debugging technique for some team members
- Have the right attitude
  - Can do, as opposed to can't be done
  - Yes, but vs Yes, and…
- Praise in public and criticize in private*
  - * Sometimes public 'floggings' have a use, but generally only to bring group back in line usually after the anti-pattern of not dealing with low performers or letting a toxic person in the group
- Be a catalyst
  - Set up the situation, accelerate a reaction
- Allow failure if it leads to growth (personal or technical)
- Be a teacher and a mentor

And even more tips

- Set clear goals
- Be honest with people (and yourself)
  - Doesn't mean tell everything though!
- Criticism Sandwiches
  - Positive, Negative, Positive
  - Serve with the right amount of bread!
    - Some people need lots of bread, others much less, but in general too much bread and the meat is likely lost or avoided
- Track Happiness
- Delegate, but get your hands dirty
- Seek to replace yourself
  - … or at least have a vacation back-up! – BUS FACTOR FOR YOU!
- Know when it is time to make waves
- Fake it until you make it…really
  - You are a phony, we all are, just don't tell anyone ☺

Tips Vary

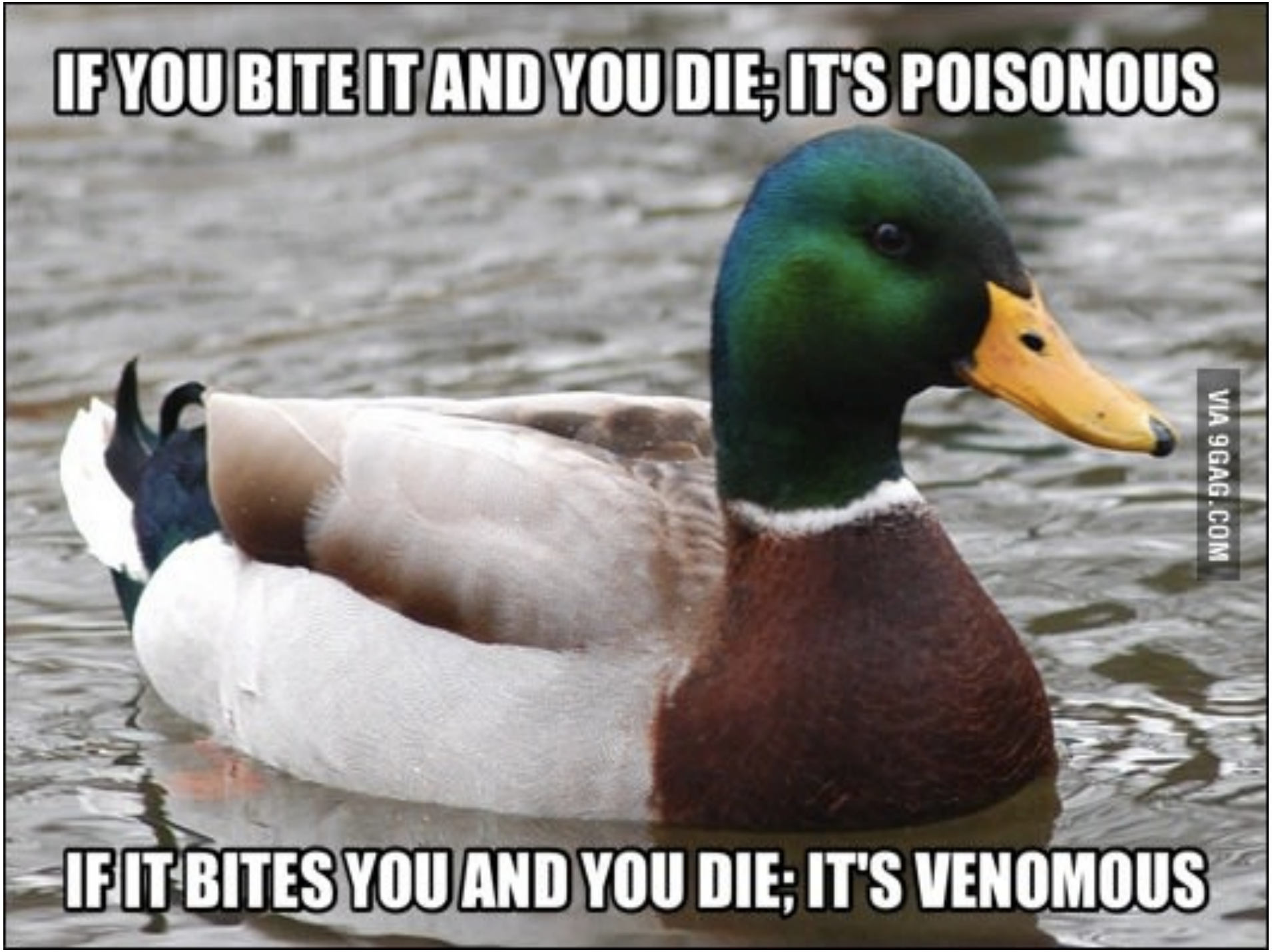
- "People are like plants" theory
  - Cactus, orchid, etc. they each have different care/feeding instructions
  - I like to think of managers more as coaches, you should try to bring the best out the players you have. Respect each and their differences and adopt your style to what works for them
- The best motivation is intrinsic motivation, though if you have a billion dollars to share with me let me know
  - Supposedly this motivation comes from giving people autonomy, mastery and purpose
    - Autonomy ability to be involved and make choices
    - Mastery ability to grow and master skills
    - Purpose make an impact
      - The purpose one gets really blown out these days with changing the world. That's great, but self-awareness needs to be practiced. A social sharing app for food pictures isn't going to change the world it may change the world of how people discuss dinner. Be precise and refine your words. Also revisit the janitor story from earlier

Manager Anti-Patterns

- Don't hire pushovers
  - Be careful with non-pushovers though they can be toxic people in disguise
- Don't ignore low performers
- Don't ignore human issues
- Understand you can't be everyone's friend
- Don't treat your team like children (nor prisoners)

Poisonous People Habits

- Lack of respecting other people's time
  - Not reading the docs before asking when they knew they could, assuming person is available on their odd schedule, etc.
- Ego
  - Overriding consensus decisions repeatedly often using negative logic and corner cases,
- Entitlement
  - Watch out here, age and generational differences abound!
- Immaturity
  - I don't agree with this one but there is a point when *SirHacksAlot's* excessive Giphy posts get on the team's collective nerves
- Paranoia
  - The "company" or "management" IS out to get you
- Perfectionism
  - See last lecture


IF YOU BITE IT AND YOU DIE; IT'S POISONOUS
VIA 9GAG.COM
IF IT BITES YOU AND YOU DIE; IT'S VENOMOUS

Anti-Venom

- Effective X% of the time where X is an integer somewhere above 0 and likely well below 100

- Don’t feed the trolls
- Check your own emotions
  - Remember self-Postel you can only control yourself
- Find the grain of truth in the venom laced bile
- Kill them with niceness
- Know when to give up
- Play the long game

Venomous Groups

- The culture of a group/organization can be poisoned as well
- We should try to change the culture if we can if not we need to get out lest it gets us
- First problem, ideal situations don't really exist at least for long!
  - Ever heard of Hedonic Adaption?
  - Things often look better from the outside
- Honestly, things tend to vary
  - Toxicity inside a good organization, oasis of sanity in bad ones
  - Toxicity can change over time, what starts out toxic may get ok with familiarity. Good things of course can be perverted as well

Practical Thoughts on Our Groups

- Team Name
- Mascot / Personality
- Team Mission Statement
- Team “By-Laws”
  - No a-holes is not a reasonable by-law folks!
- T-Shirts, stickers and group focused swag might just be digital this quarter
- Getting to know the people beyond the code
- Developing the witty back and forth (…or not)
  - Professional code soldiers aren’t necessarily throwing off one liners like Arnold Schwarzenegger as they slaughter bugs
- War Stories* - come later

Coming Up in CSE 112

- This week
  - Team Design and Build
  - The Motivating Premise of the quarter
  - The idea of the “Pitch”
  - First pipeline and process (the very simple version) will start