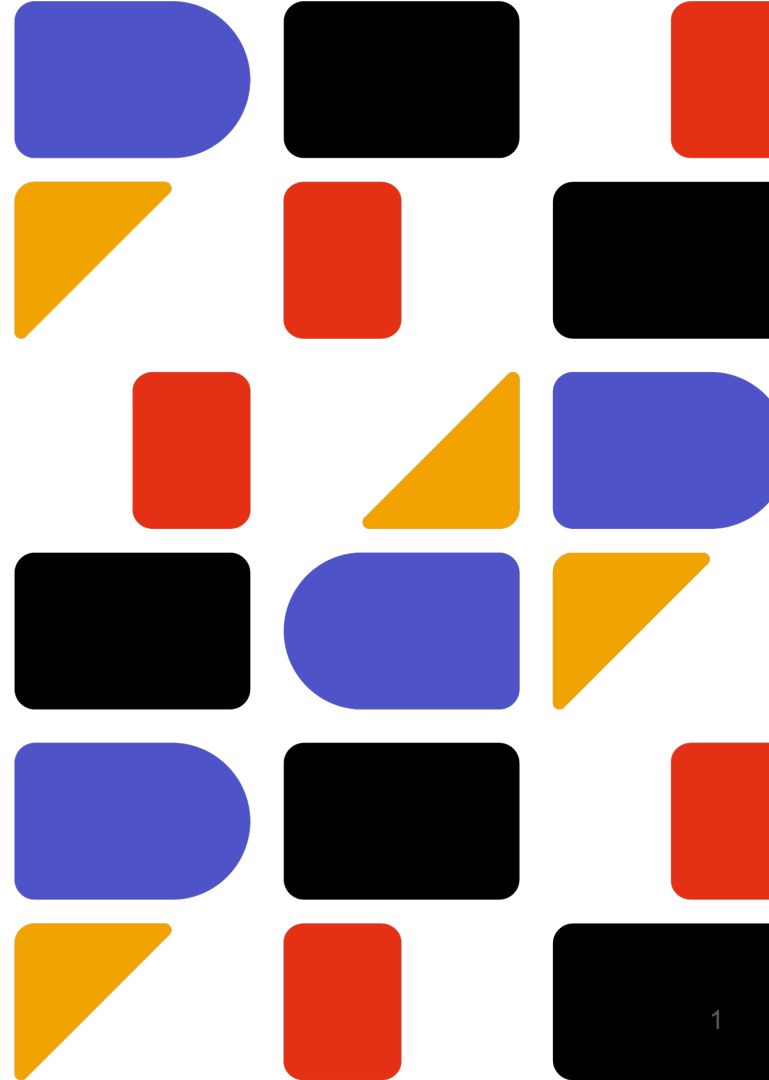


Laziness at Scale



SPEAKERS



Adam McCormick
Software Engineer
Meta

Agenda

2:00-2:05

Meet the Speaker

2:05-2:20

Engineering Mindset

2:20-2:35

Dealing with Management

2:35-2:50

Saying “No”

2:50-3:00

Angry Heckling (Q&A)

Engineering Mindset

Thinking without so much thinking

01

Engineering is not Science

Understanding is key to finding your niche



The odds are greatly against you being immensely smarter than everyone else in the field

– Akin's Laws of Spacecraft Design

How Science Works

- Begin with observation
- Define rules that match the knowns
- Test your rules
- Repeat ad infinitum

“Science improves the definition of mediocre”



How Engineering Works

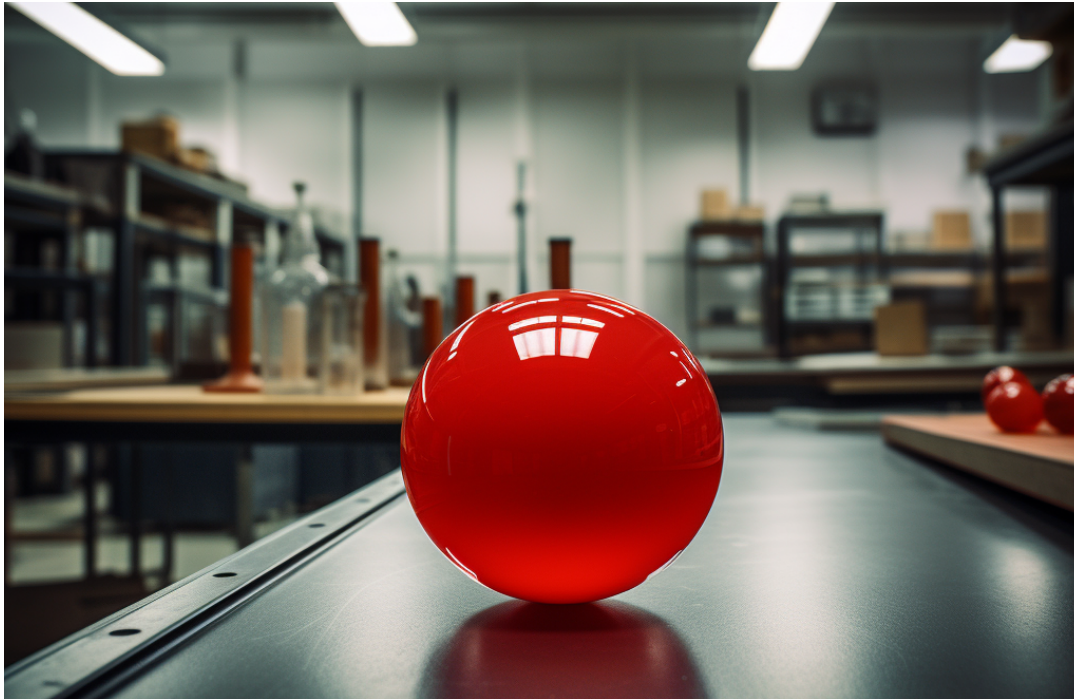
- Begin with an unsatisfied requirement
- Establish a mechanism within tolerances
- Measure system with mechanism in place
- Repeat as needed

“Engineering makes mediocre cheaper”



Story Time - The Red Ball (A Parable)

A Scientist, a Mathematician, and an Engineer are all tasked with finding the volume of a red ball



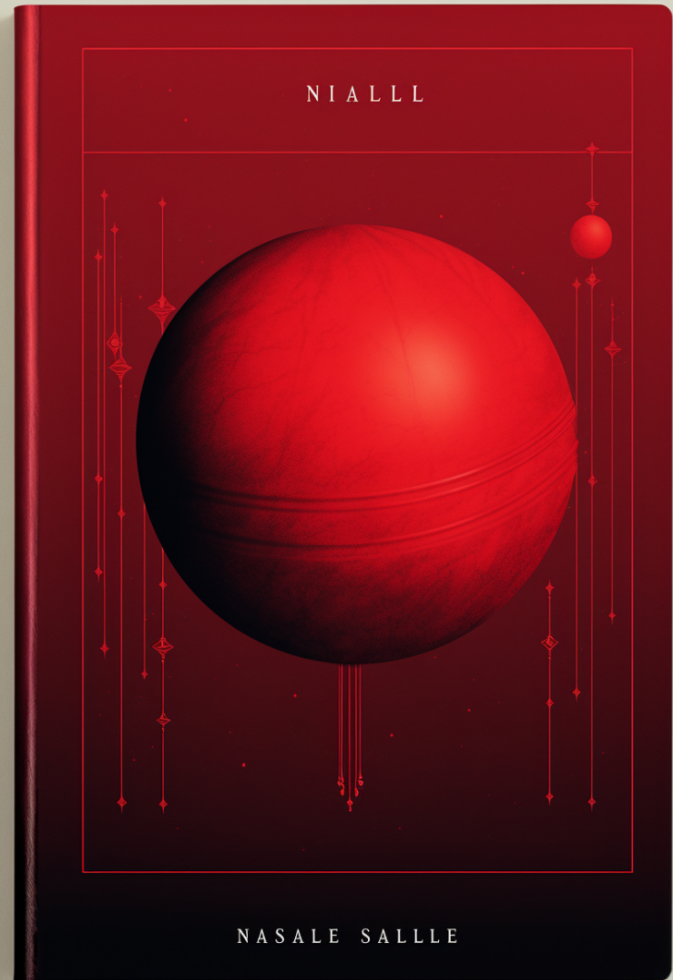
The Mathematician calculates the volume from the size



The Scientist measures the displacement



The Engineer looks up the serial number in the Red Ball Reference Manual



The Lesson - Don't reinvent, adapt

Instead of:

- Clever solutions
- Writing from scratch
- New technology (Resume-driven dev)
- Original research
- Radical modifications

Focus on:

- Learning effective patterns
- Adapt existing solutions
- Well-supported technology
- Meta analysis
- Minimum working change, then repeat

02

Engineering is Concrete

Build assuming you'll have to prove it works



Engineering is done with numbers. Analysis without numbers is only an opinion.

– Akin's Laws of Spacecraft Design

Story Time - Background Restricted Mode

- An executive was shown a notice that a flagship application was being restricted
- A V-Team was build including DS, Engineering, Partnerships, Management

What went wrong:

- No goal or deliverable was defined
- Driven by executive instinct
- An acceptable level of user pain not defined
- No indication that the problem is solvable

Why it wasn't worse:

- Data was available to describe the impact and bound the problem
- Partnerships already established with OEMs

03

Engineering is Lazy

Overworking considered harmful



There's never enough time to do it right, but somehow,
there's always enough time to do it over.

– Akin's Laws of Spacecraft Design

Story Time - Jupyter Notebooks

- Ad-hoc analysis was done in a system that wasn't compatible with ongoing data pipelines
 - Different SQL flavor, processing language, security framework, operational model
- Same problem for dashboards and various hosted workflows
- Adding functionality like privacy controls and auditing required duplication

<https://engineering.fb.com/2023/08/29/security/scheduling-jupyter-notebooks-meta/>

HOW STANDARDS PROLIFERATE: (SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)



Story Time - Jupyter Notebooks (cont...)

What could have happened:

- New product that does everything
- Base workflow on the data pipeline and retrain engineers
- Start from scratch interface that does everything

What we did instead

- Repurposed a working UI from one of the most-used systems
- Migrated a piece at a time then cut off “Legacy” tooling
- Made the solution pluggable for various use cases

<https://engineering.fb.com/2023/08/29/security/scheduling-jupyter-notebooks-meta/>

The Lesson - Prefer Evolutionary Change

Instead of:

- Building products
- New implementations
- Endless tweaking before release
- Big projects and bigger releases
- Handling every contingency

Focus on:

- Building into existing systems
- Harvesting existing assets
- Build, release, measure, pivot
- Small, discrete, contained changes
- Solve the problem at hand

Dealing with Management

The fine art of being useful

01

Managers Have Managers

Avoiding the crossfire of organizational politics



[S]ed quis custodiet ipsos custodes?
(Who will watch the watchers?)

– Satire VI, Decimus Junius Juvenalis

Sed quis curatores administrabit? (Who will manage the managers?)

– Me and Google translate (don't @ me)

Story Time - Goldfish Management

- Manager presses for specific direction in 1:1
- Requires status reports, plays Product Owner, drives a delivery timeline
- IC makes the changes, works with partners, delivers the request
- Manager's stakeholders do not want the deliverables
- Manager has now miraculously never heard of this project
- IC's time wasted, No impact for review time

The Lesson - Don't Get Steamrolled

Instead of:

- Taking one voice as truth
- Viewing projects in isolation
- Jumping on every problem

Focus on:

- Test the waters on the need
- Getting the bigger picture
- Solving the right problems

02

Managers are Human

Relationships matter more than hierarchies



The Forever Manager

- Usually “worked their way up” to the role
- Likely has turned down pushes to advance
- Unflappable in times of crisis
- Knows all the rules, bends them as needed

Getting the Most From Them

- Over-communicate by default
- Seek advice, not decisions
- Prioritize responding to questions
- Can tell you where the bodies are buried



The Reluctant Manager

- Typically a senior engineer pressed to lead
- Continually drifts into the details of the work
- Will try to speak for the team
- May be defensive about their role

Getting the Most From Them

- Negotiate and collaborate
- Include them in discussions
- Lead with the challenge, explain the solution
- Make all requests direct and explicit



The Empire Builder

- Typically younger and specialized
- Never agrees to anything explicitly
- Always willing to listen
- Trusts you, expects trust

Getting the Most From Them

- Focus on defining and articulating scope
- Refer external asks to them
- Editorial first, details after
- Volunteer when they ask



The Tyrant

- Typically middle-management
- Hyperfocused on deadlines
- Wants reports more than results
- Process is the priority

Getting the Most From Them

- Feed them options you agree with
- Always give them a way to save face
- Don't reward interruptions
- Communicate in tradeoffs not ultimatums



The Executive

- Typically a career employee or founder
- Serious 120% of the time
- Can always make a decision (or decide how)
- Is right more often than they are wrong

Getting the Most From Them

- Give them solid numbers
- Bring realistic options solving real problems
- Don't suggest anything you can't accomplish
- Include them when you need finality



03

Managers Need Support

The best relationships come from complementing each other



Not having all the information you need is never a satisfactory excuse for not starting the analysis.

– Akin's Laws of Spacecraft Design

Story Time - ANR Instrumentation

- App Not Responding is a type of Android crash common in large apps
- Notoriously hard to observe because sometimes the app “just recovers”
- Demands from leaders to get visibility for specific issues they saw in Google Play

What we did first:

- Adding tracking to known problematic methods
- Complex server-side processing to present
- Not compatible with other Crash analysis tools

What worked better:

- Holistic monitoring using core Android looping construct
- Recategorization based on deep analysis of crashes
- Deep integration with existing tooling

The Lesson - Good Enough, then Better

Instead of:

- Future-proofing
- Building against the knowns
- Starting from the ask
- Polished Tooling
- Feature-driven development

Focus on:

- Build when the gap is known
- Understand the need
- Solving user pain
- Incremental functionality
- Problem solving

Saying “No”

Or how you should learn to stop worrying and embrace the minimum

01

Busy is Waste

Do the work that needs doing, and only that



When in doubt, estimate. In an emergency, guess. But be sure to go back and clean up the mess when the real numbers come along.

– Akin's Laws of Spacecraft Design

Story Time - Crash Triage

- Meta has a Monorepo that covers all out mobile and VR products
- Hundreds of thousands of builds a day
- New crashes appear every day
- Our group deals with crash reliability so the question is always to triage or fix
- Triageing is a question of communication, research, and evangelism
- Fixing is putting your headphones on and burning down the list of crashes

Triage wins because:

- There are thousands of people introducing and fixing defects every day
- Triage is a specialized skill not shared by the people introducing the crashes
- Putting the crash in the right hands prevents duplicated effort

The Lesson - Specialization Beats Effort

Instead of:

- Going heads-down on every task
- Continual movement
- Being a cog in the machine

Focus on:

- Specialize and grow into expertise
- Steady progress
- Be a lever

02

Be Realistic

Make the best of whatever slog happens



[In Engineering,] there's no justification for designing something one bit "better" than the requirements dictate.

– Akin's Laws of Spacecraft Design

Story Time - Kotlin Conversion

- Early in 2020 Meta began adoption of Kotlin
- 10 Million lines converted over first 30 months

What went wrong:

- Automated conversions waited on massive rework of DI, Test, and other infrastructure
- Tens of site outages, thousands of man hours, huge expenditure of compute time

Why it wasn't worse:

- Interoperability from the start
- Clear milestones and pivot points
- Automation at the forefront (converters, alerts, tests), manual as an exception

<https://engineering.fb.com/2022/10/24/android/android-java-kotlin-migration/>

The Lesson - Don't go looking for problems

- The cost for rewrites and overhauls is almost always higher than the payoff
- Rewriting a system will mean you lose and have to relearn tribal knowledge
- Big conversions are never really “done,” have to think in half-lives
- Time is better spent addressing dev pain than hoping a rewrite will fix everything

If you must rework a system:

- Automate as much as you can
- Maintain integration with production system
- Front-load critical systems
- Testing and Alerting can be your friends

03

Communicate

Choosing the right words and playing politics



A bad design with a good presentation is doomed eventually.
A good design with a bad presentation is doomed immediately.

– Akin's Laws of Spacecraft Design

Participate

- Try to speak at least once in every meeting
- Make sure your contribution is relevant
- Avoid being the first to speak
- “Strong man” good ideas and give credit
- Don’t state the obvious
- Don’t minimize yourself

This avoids seeming disengaged or domineering



... or Don't Attend

- Only join a meeting as a contributor
- Limit attendance in technical discussions
- Consider written status over meetings
- Prefer remote attendance if in doubt

This minimizes wasted time and gets more done



Avoid Assigning Fault

- “That turns out not to be the case”
- “Upon further examination we found”
- “Circumstances have changed”
- “That assumption proved inaccurate”

This keeps people from getting defensive



Take Responsibility not Credit

- “We built...”
- “Our consensus is...”
- “The Team feels...”
- “I didn’t see that...”
- “I will look into...”
- “I can help you with...”

This makes it sound like you are an authority



History is Written by the Victors


- Get good at writing prose
 - Not taking notes or writing documentation
 - Editorialize, but support with data
 - Avoid boilerplate updates
- Write about what you're doing anyway
 - Summarize your problems
 - Talk about challenges
 - Include intermediate findings
- Publish your work
 - Make sure a broad audience can weigh in
 - Engage with the audience
- Reference previous work
 - Both yours and those you want to sway



WHERE CITATIONS COME FROM:

CITOGENESIS STEP #1:
 THROUGH A CONVOLUTED PROCESS,
 A USER'S BRAIN GENERATES FACTS.
 THESE ARE TYPED INTO WIKIPEDIA.


THE "SCROLL LOCK" KEY WAS
 DESIGNED BY FUTURE
 ENERGY SECRETARY STEVEN
 CHU IN A COLLEGE PROJECT.



STEP #1

**A RUSHED WRITER CHECKS WIKIPEDIA
 FOR A SUMMARY OF THEIR SUBJECT.**

US ENERGY SECRETARY STEVEN CHU,
 (NOBEL PRIZEWINNER AND CREATOR OF
 THE UBIQUITOUS "SCROLL LOCK" KEY)
 TESTIFIED BEFORE CONGRESS TODAY...




STEP #2

**SURPRISED READERS CHECK WIKIPEDIA,
 SEE THE CLAIM, AND FLAG IT FOR REVIEW.
 A PASSING EDITOR FINDS THE
 PIECE AND ADDS IT AS A CITATION.**

GOOGLE IS YOUR
 FRIEND, PEOPLE.

<REF>{{CITE WEB|URL=



STEP #3

STEP #4
 NOW THAT OTHER WRITERS
 HAVE A REAL SOURCE, THEY
 REPEAT THE FACT.

MORE
 CITATIONS

SLIGHTLY
 CAREFUL
 WRITERS


BRAIN

WIKIPEDIA

CITED
 FACTS

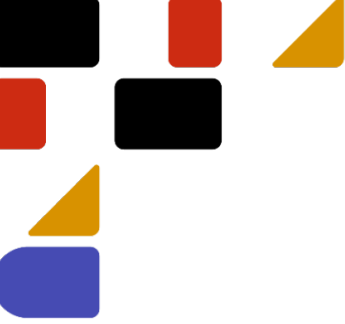
CARELESS
 WRITERS

REFERENCES
 PROLIFERATE, COMPLETING
 THE CITOGENESIS PROCESS.



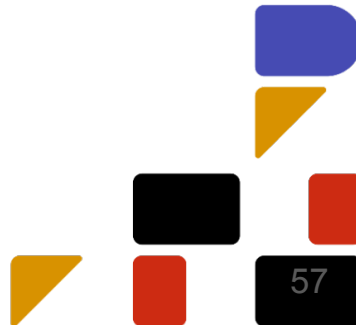
The great enemy of communication, we find, is the illusion of it.

– *Is Anybody Listening?* - William H. Whyte



Questions? Answers?

Angry Heckling?





THANK YOU!

A presentation slide for Adam McCormick's talk. The slide has a black background with a portrait of Adam McCormick on the left. The text on the slide includes the event name, location, dates, the title of the talk, and the speaker's name and title. The Meta logo is also present in the bottom left corner of the slide.

 DPE
SUMMIT

San Francisco, CA
September 20-21

Laziness at Scale

 **Adam McCormick**
Software Engineer
Meta