



# Keep Calm and Carry On:

Scaling Your Org To Deliver Great Software

Charity Majors, @mipsytipsy







# Keep Calm and Carry On:

Scaling Your Org To Deliver Great Software

Charity Majors, @mipsytipsey







# **Growing up is hard to do**

The latest parenting trend for software engineering teams

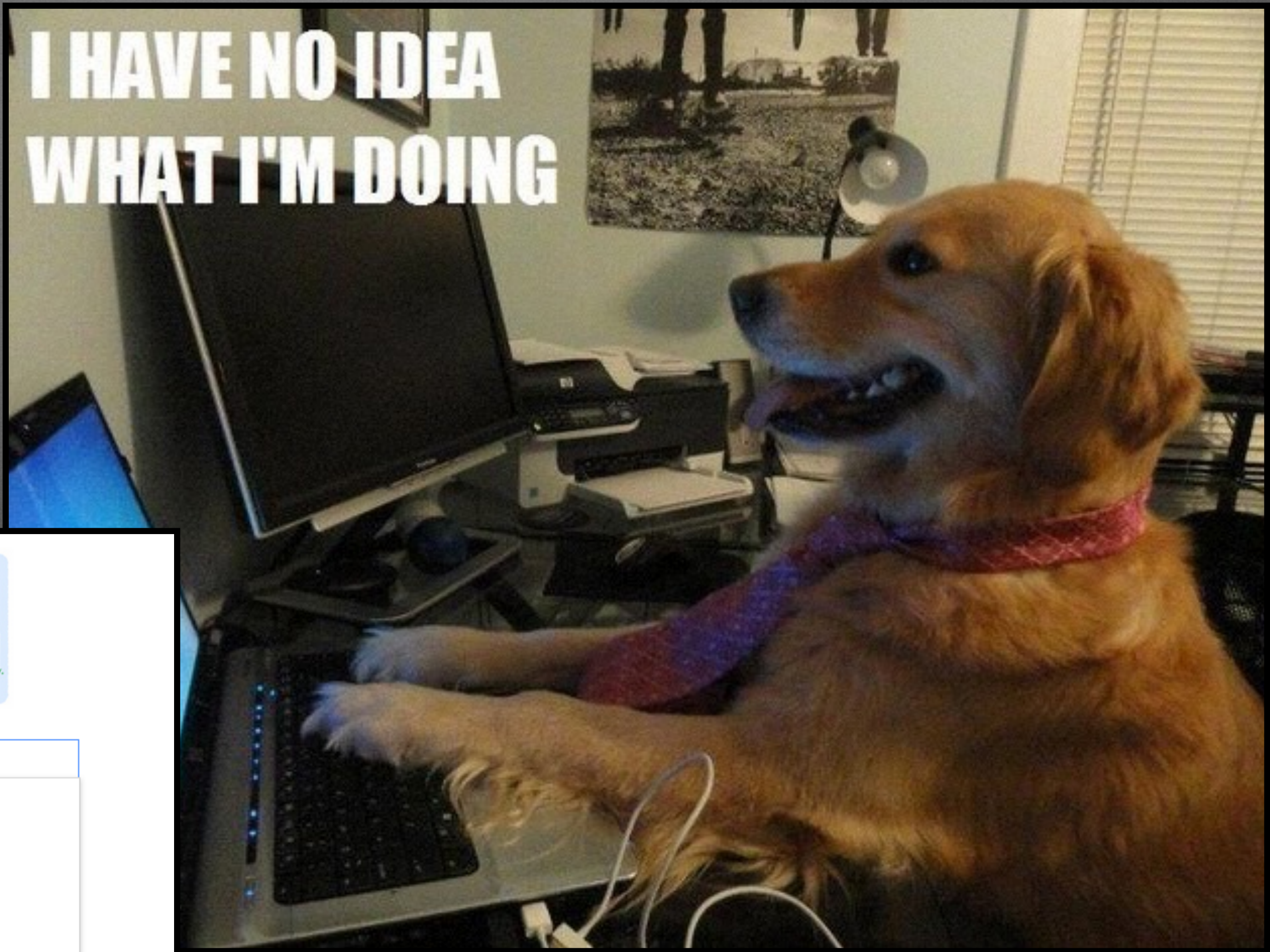
## **How to fail at microservices**

And life in general. Before you ever even start!

### **Failing failures and fail-ers who fail at them**

With software.





what is a microse

what is a **microservice**

what is a **microsecond**

what is a **microserver**

what is a **microservice in java**

what is a **microservice example**

what is a **microservice architecture**

what is a **microsecond to millisecond**

what is a **microsite**

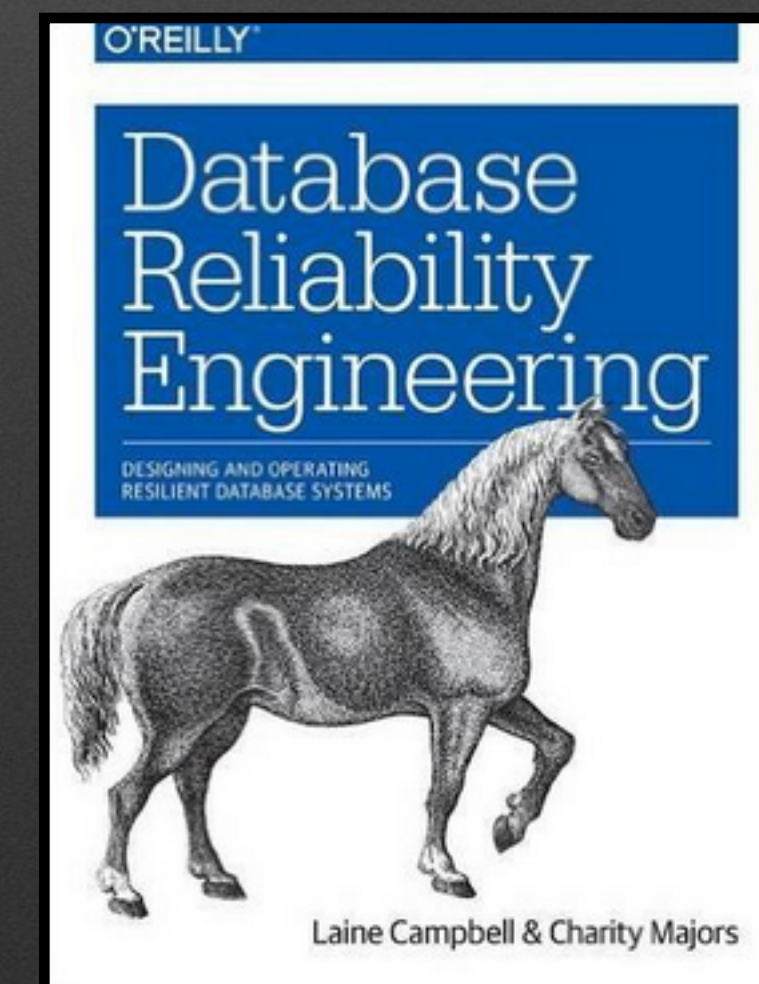
what is a **microserver used for**

Google Search    I'm Feeling Lucky





@mipsytipsy  
engineer, cofounder, CTO








## Some predictable organizational effects of microservices:

- Conway's Law
- Swap tech problems for political
- Multiple repos
- On-call burnout
- Distributed monoliths
- Software engineers responsible for services







 **Charity Majors** @mipsytipsey · Oct 29

if you ever tried to roll out microservices and it turned into a tire fire, DM the details. will safely anonymize. 🦄🍸

← ↻ 2 ❤️ 10 || ⋮

“Dear Twitter ...”





“Software deploys ... that take days to run, when they run.”



“I’m responsible for it, but I can’t log in to it.”



Hard things are hard.





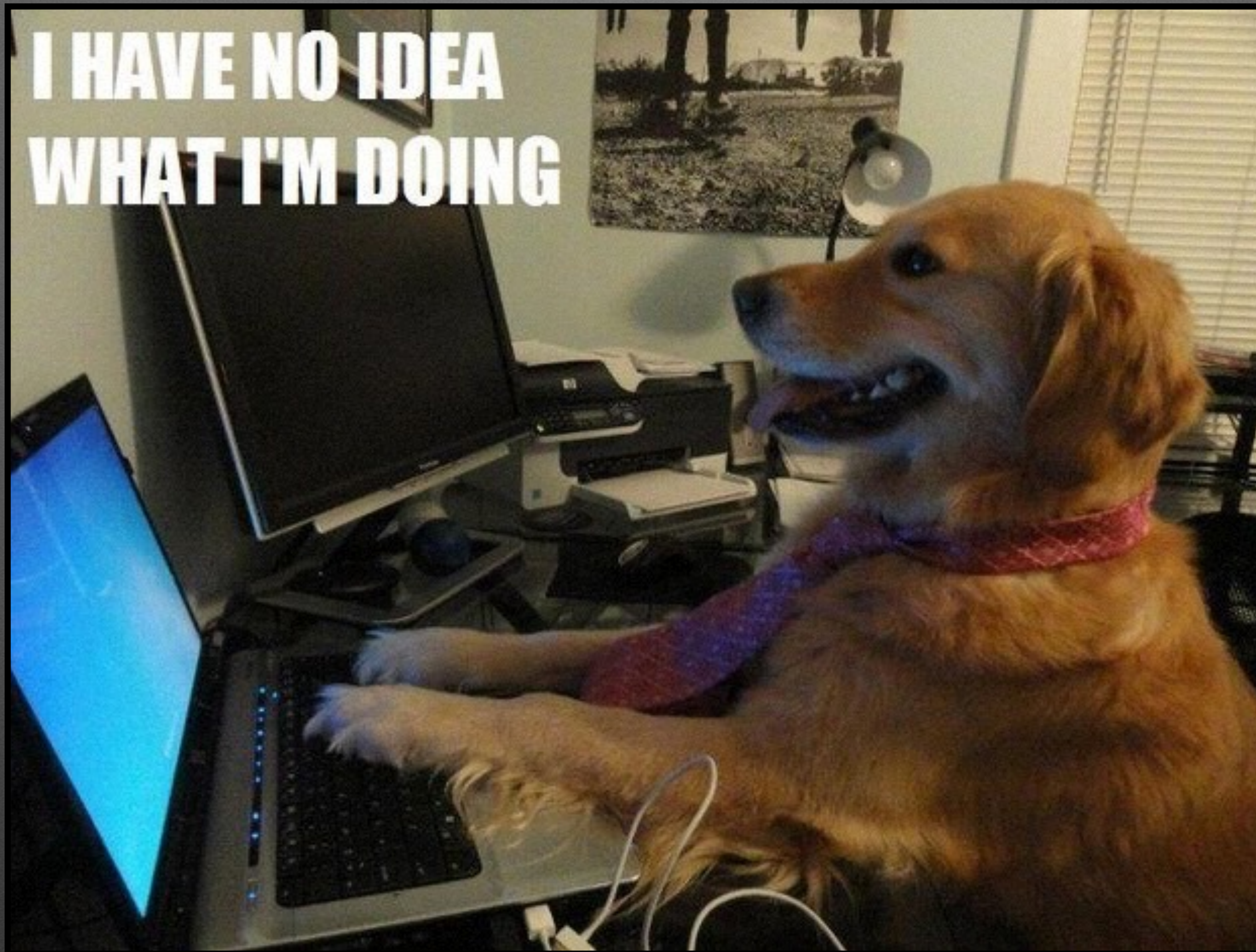
**Don't try and tell me how to solve  
my people problems...**



**you probably can't. and I probably  
don't know how to listen.**

**Give me stories. and tools.**





**“What’s a microservice?”**

*~me*



# What are microservices?



FANART © TIFFANY ([HTTP://SELINMARSOU.DEVIANTART.COM](http://selinmarsou.deviantart.com))  
APPLEJACK © HASBRO

- Monorepo — sometimes
- Independently deployable, small modular services
- Decentralized governance
- Small teams, up to maybe a dozen people
- Operating independently, interacting with other teams via APIs



**Microservices are about changes.**





*Microservices are our latest experiment to recreate the terrific speed, autonomy, and productivity of early startup teams ... at big and growing companies.*







# can i haz microservices?

- Team structure (Conway's Law?)
- Communication pathways
- "Smarter Edges": For individual contributors
- "Dumb Pipes": for managers
- Transitions are hard



# YAS! Has microservices: just the good parts

- Don't get religious. It's not all or nothing.
- What are your team's strengths? What are their weaknesses?
- Account for the operational cost





How many engineers do you have?

How good are they at operations?

**\*\* you need to be REALLY GOOD at operations to do microservices.**



WE GOT RID OF OUR OPS TEAM BECAUSE WE DECIDED  
SOFTWARE ENGINEERS SHOULD BE ON CALL.  
SO ALL THE SOFTWARE ENGINEERS WERE  
GETTING WOKEN UP NIGHT AND DAY, AND THEY  
STARTED QUITTING. IT WAS A DEATH SPIRAL  
WHEN I NOPED OUT OF THERE.





screenshot of databases, stateless

How many products/services do you really have?

Use a big fat service if it helps, plus some smaller ones

Don't microservice your shared libs, storage, or registry

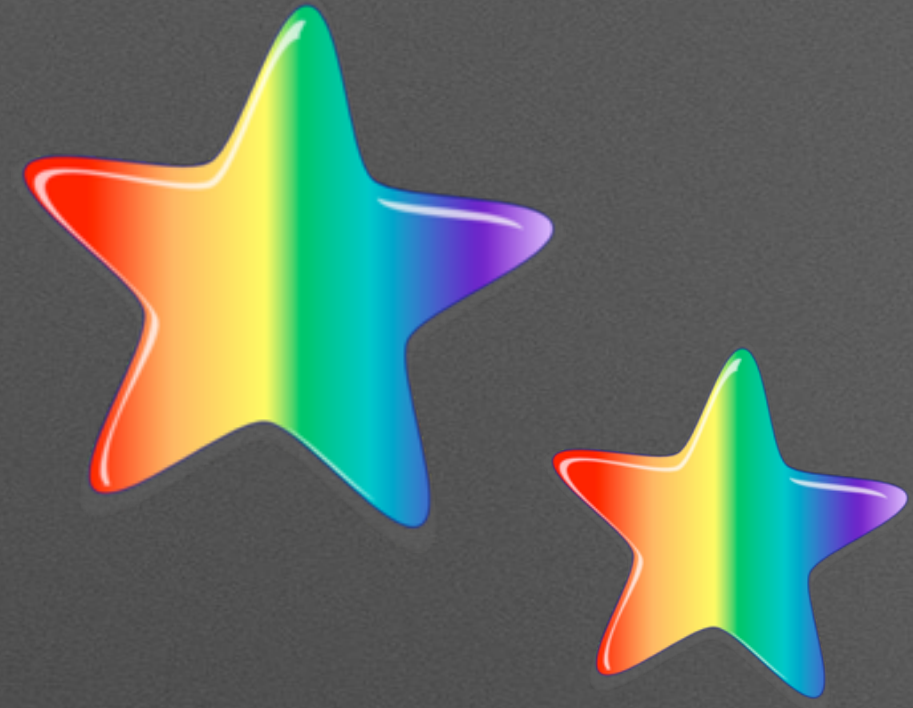


**Don't reinvent too many wheels.**

new wheels have too many unknown-unknowns

(“choose boring technology”: still applies)





# Operability / Teams.

- The mission
- Build a cult (j/k) (no really)
- Let your team innovate.



THE DBAS DIDN'T TRUST US TO TOUCH THE DATABASES.  
SO THEY WERE THE ONLY ONES WITH PASSWORDS,  
EVEN THOUGH WE WERE ALL ON CALL  
FOR OUR OWN SERVICES.

SOMETIMES THEY JUST WOULDN'T WAKE UP.  
WE HAD TO SIT THERE HELPLESSLY WHILE  
HALF A DOZEN SERVICES WERE DOWN.





THE SWES DIDN'T WANT TO UNDERSTAND WHAT WAS HAPPENING,  
THEY JUST WANTED A RECIPE TO 'FIX IT' WHEN A DB GOT SLOW.

THEY WOULD RUN COMMANDS FROM MY SHELL HISTORY  
WITHOUT HAVING ANY IDEA WHAT IT WOULD DO!

and from a DBA at a different company ... ..





**We *\*must\** pair responsibility with  
empowerment.**



WE HAD STARTED OUT WITH LIKE 4-5 SERVICES BUT GREW FAST.  
HAD 20 OR 30 MICROSERVICES BY THE TIME I LEFT.

OPERATIONAL COMPLEXITY WENT WAY UP WHILE TEAM SHRUNK.  
I WAS THE ONLY OPS GUY FOR ALL OF IT.



**Have you considered ... valuing non generalist  
SWEs and their work?**



THERE WAS A BREACH ELSEWHERE IN THE INDUSTRY. MY BOSSES  
OVERREACTED AND LOCKED DOWN MORE THAN THEY NEEDED FOR PCI.  
NOBODY HAD THE ACCESS THEY NEEDED TO GET WORK DONE.

NETWORK FOLKS WERE IN ANOTHER OFFICE.  
WE FELT LIKE SECOND CLASS CITIZENS.





THE BIG CHALLENGE WAS NETWORK/FIREWALL STUFF. THINGS WERE SO LOCKED DOWN, TWO NODES ON THE SAME VLAN COULDN'T COMMUNICATE ON A GIVEN PORT.

CHANGES WERE UNANNOUNCED, THERE WAS NO WAY OF PREDICTING THEM OR GETTING A LOG OF WHAT HAPPENED.

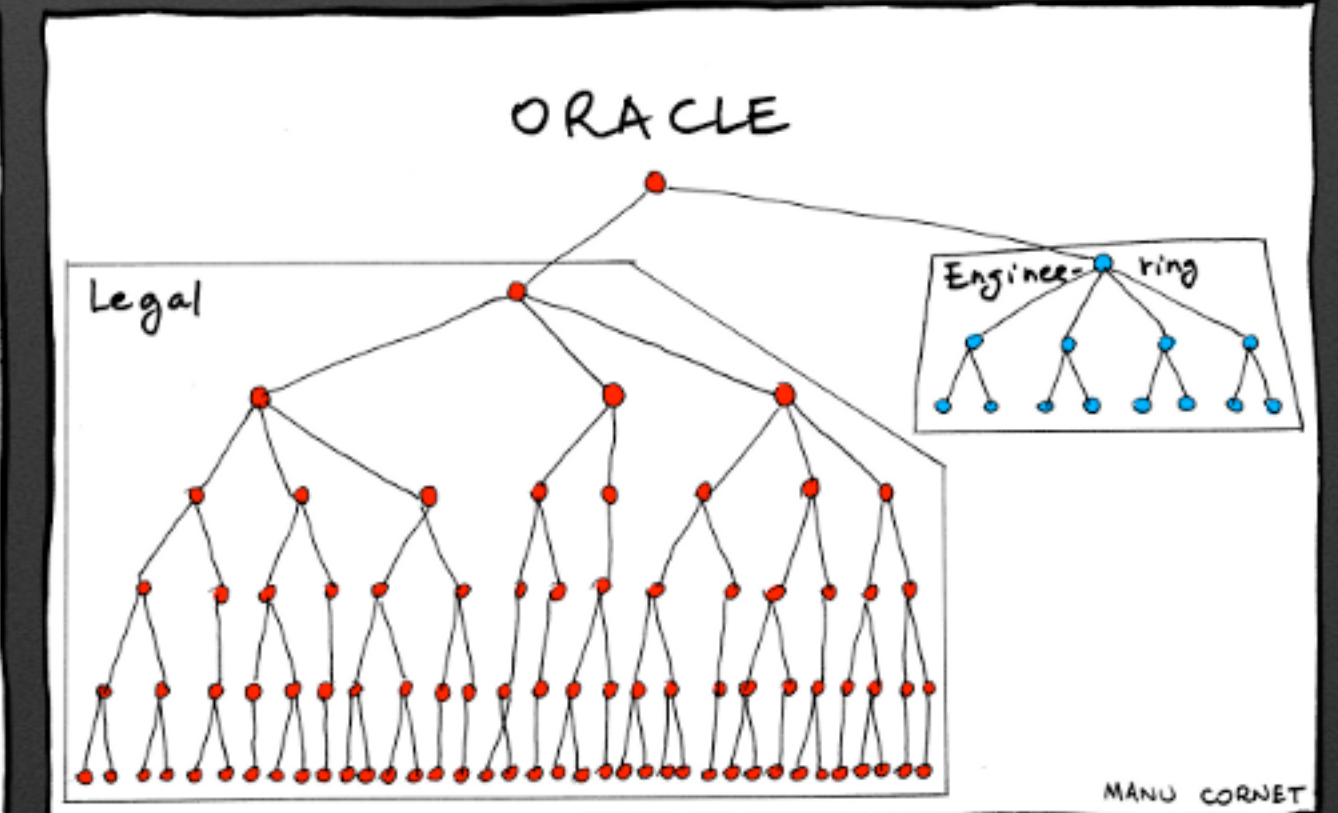
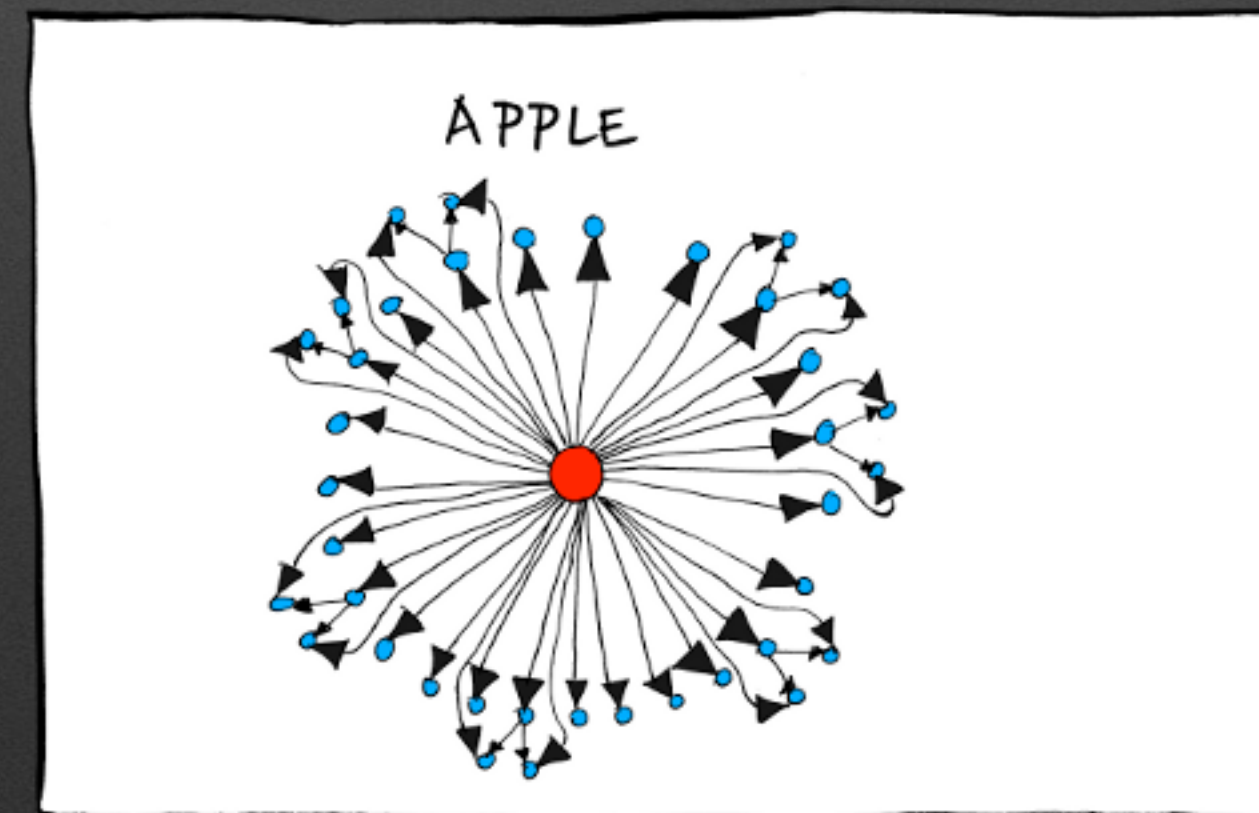
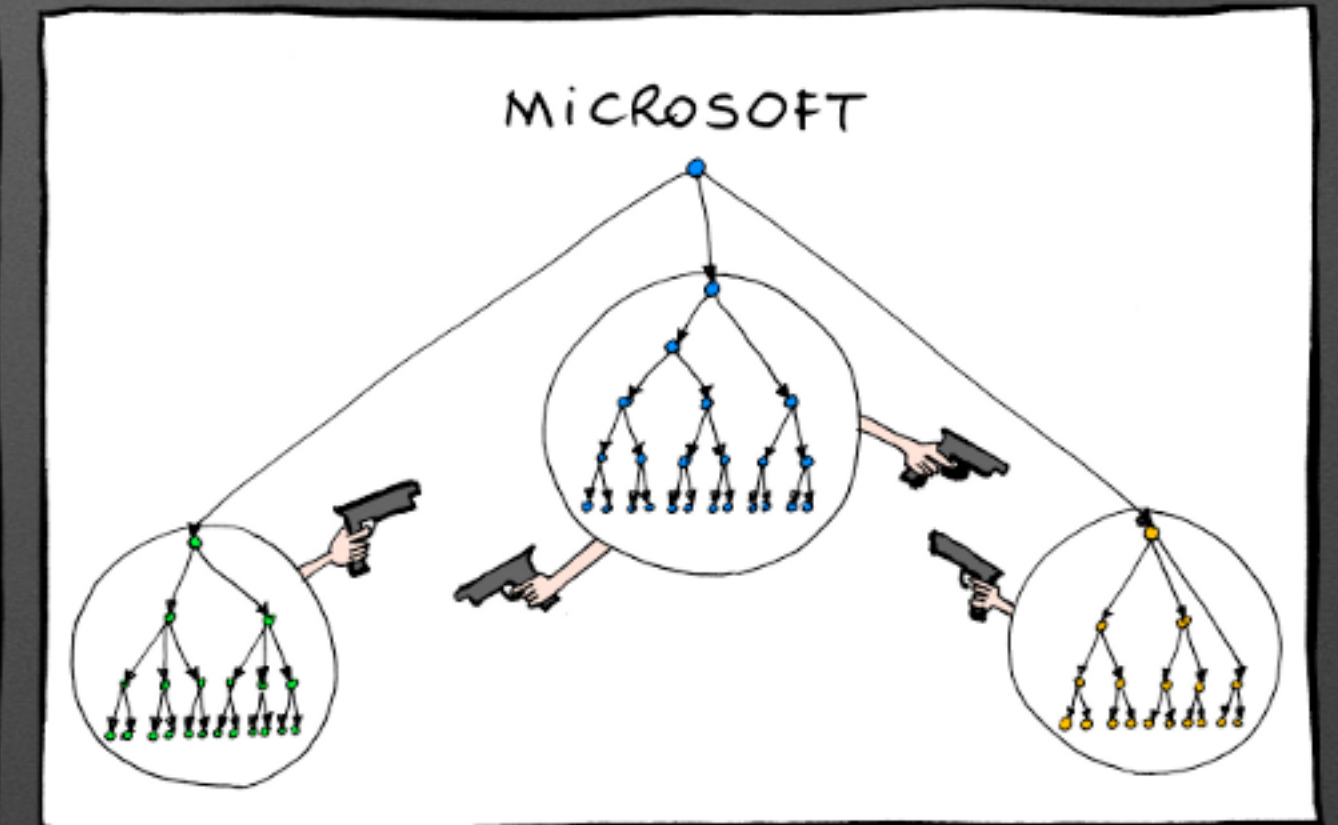
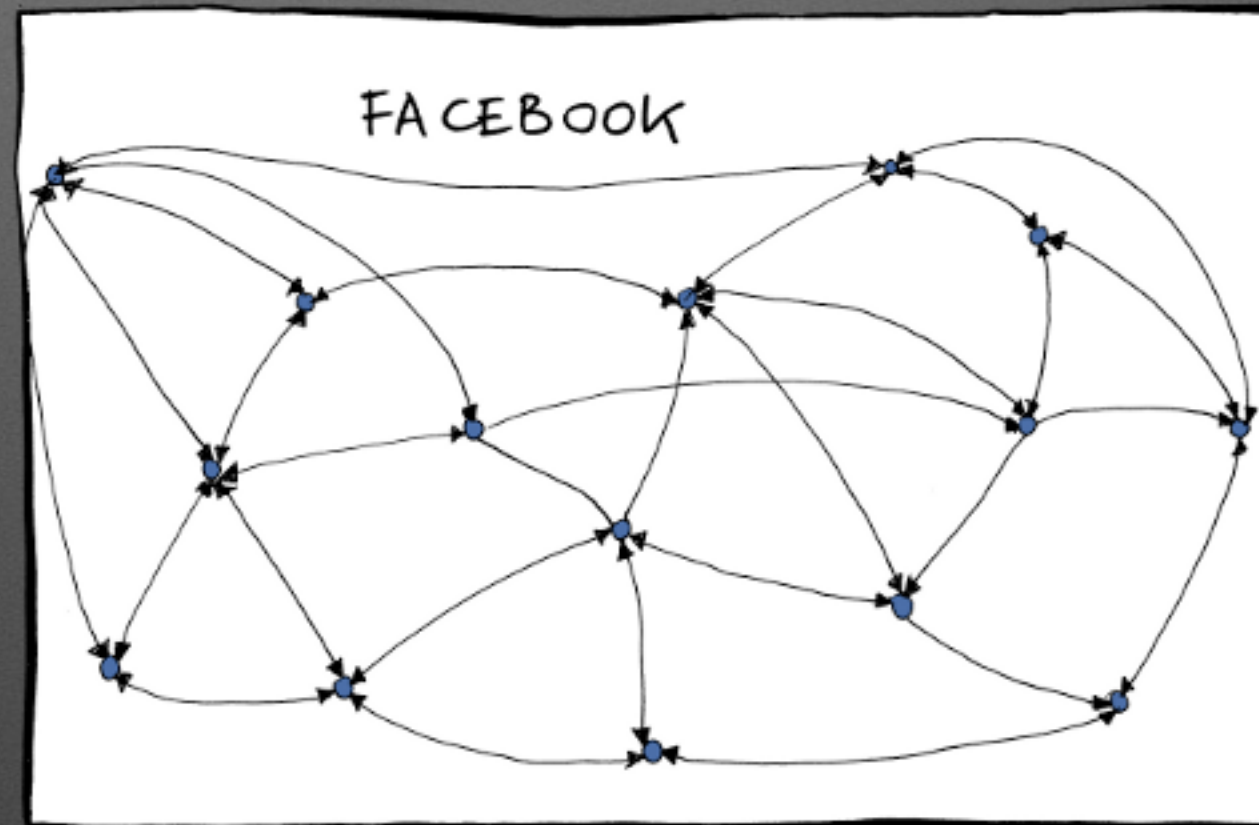
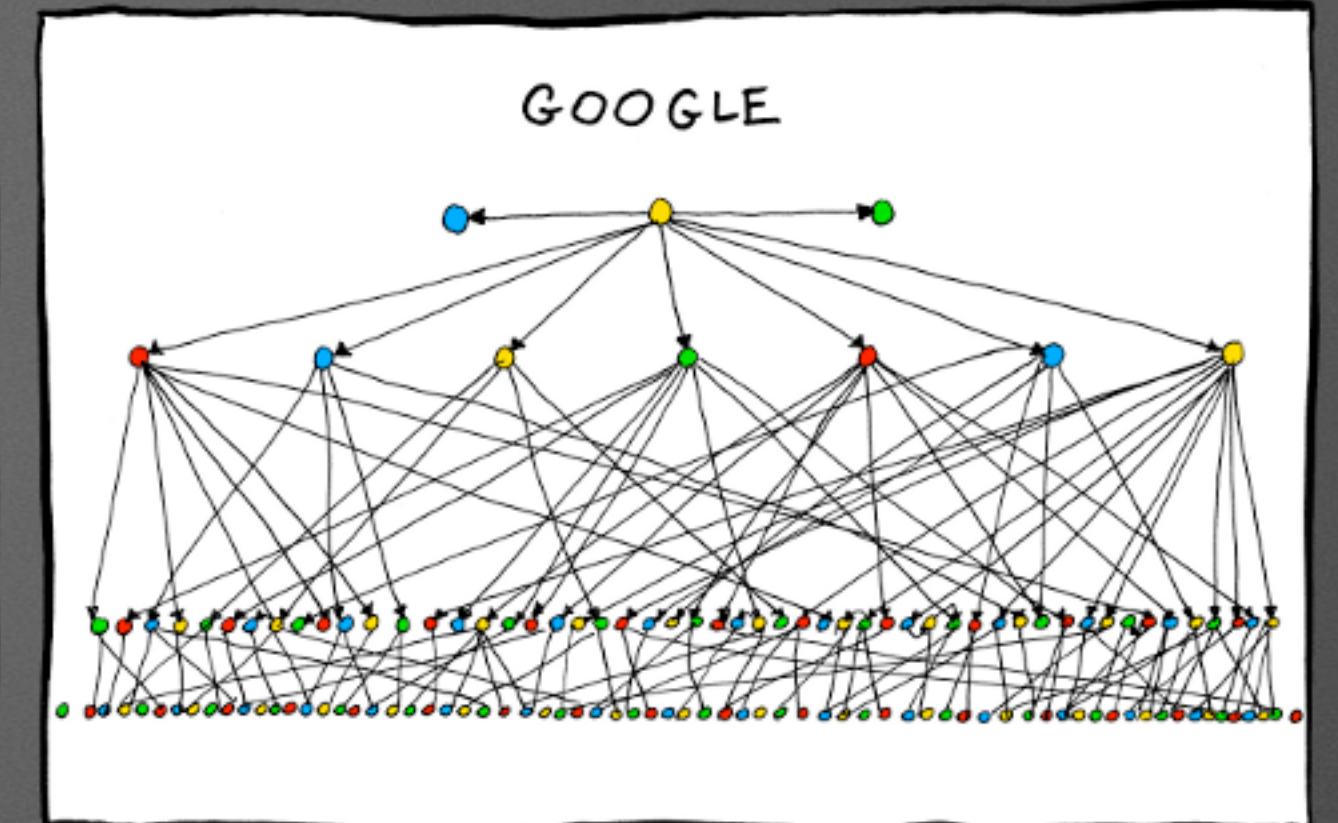
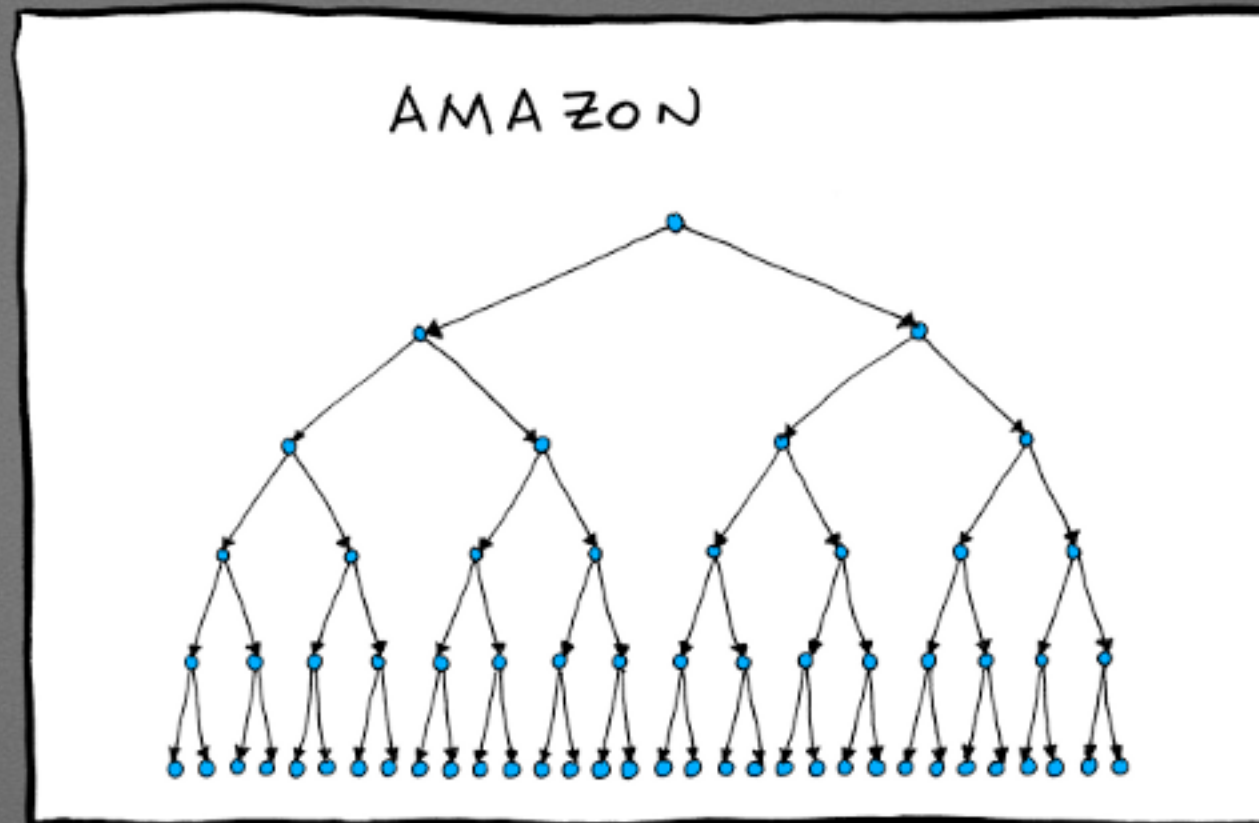
AND SOMETIMES OOPS, CHANGES WOULD REGRESS SOMEHOW AND NOW SERVICE X COULDN'T TALK TO Y AND MESSAGES ARE BUILDING UP ON THE MESSAGE BUS. WE COULDN'T REACH THE TEAM AND HAD NO ACCESS TO INVESTIGATE OURSELVES.



networking: common theme

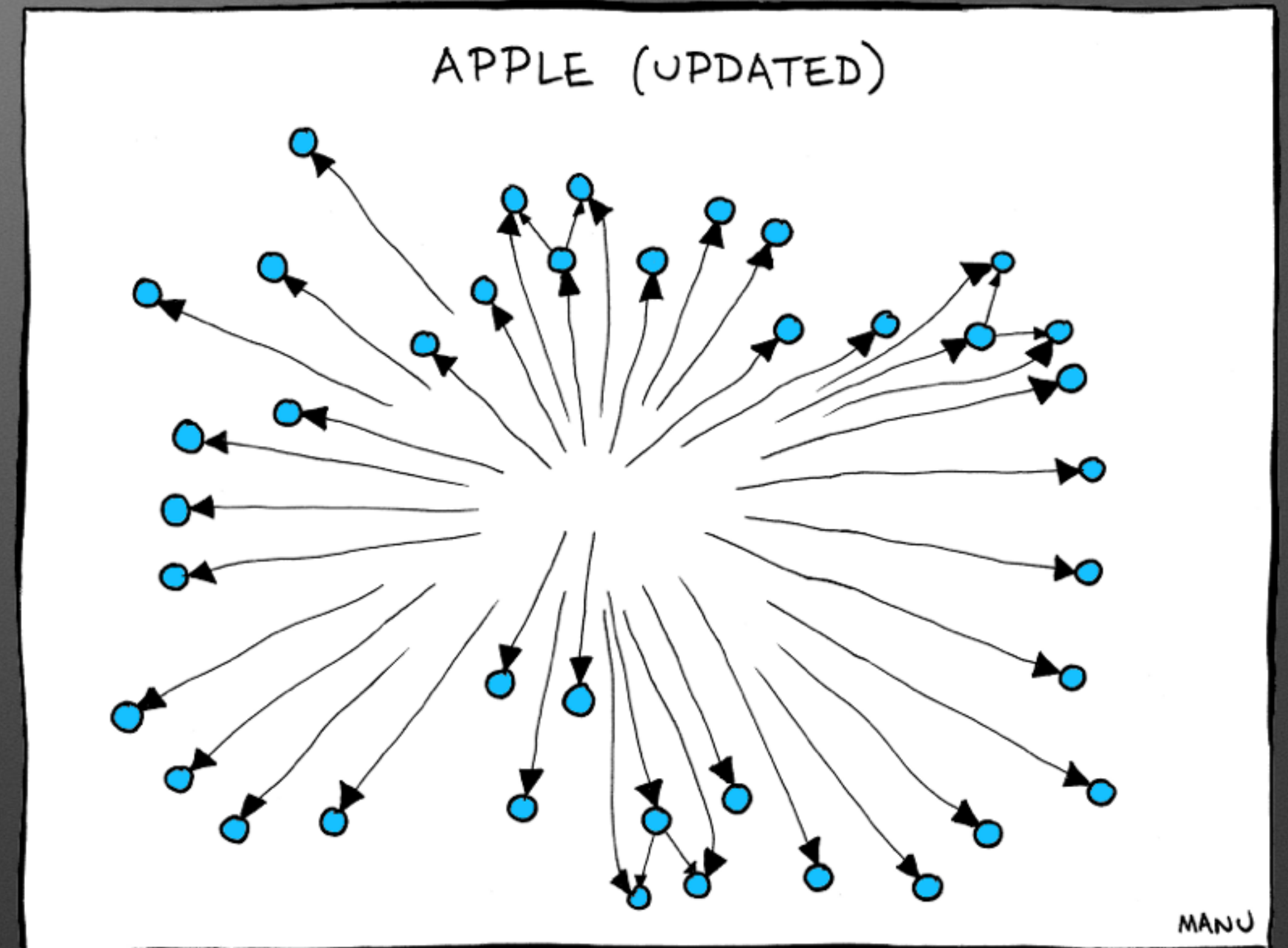
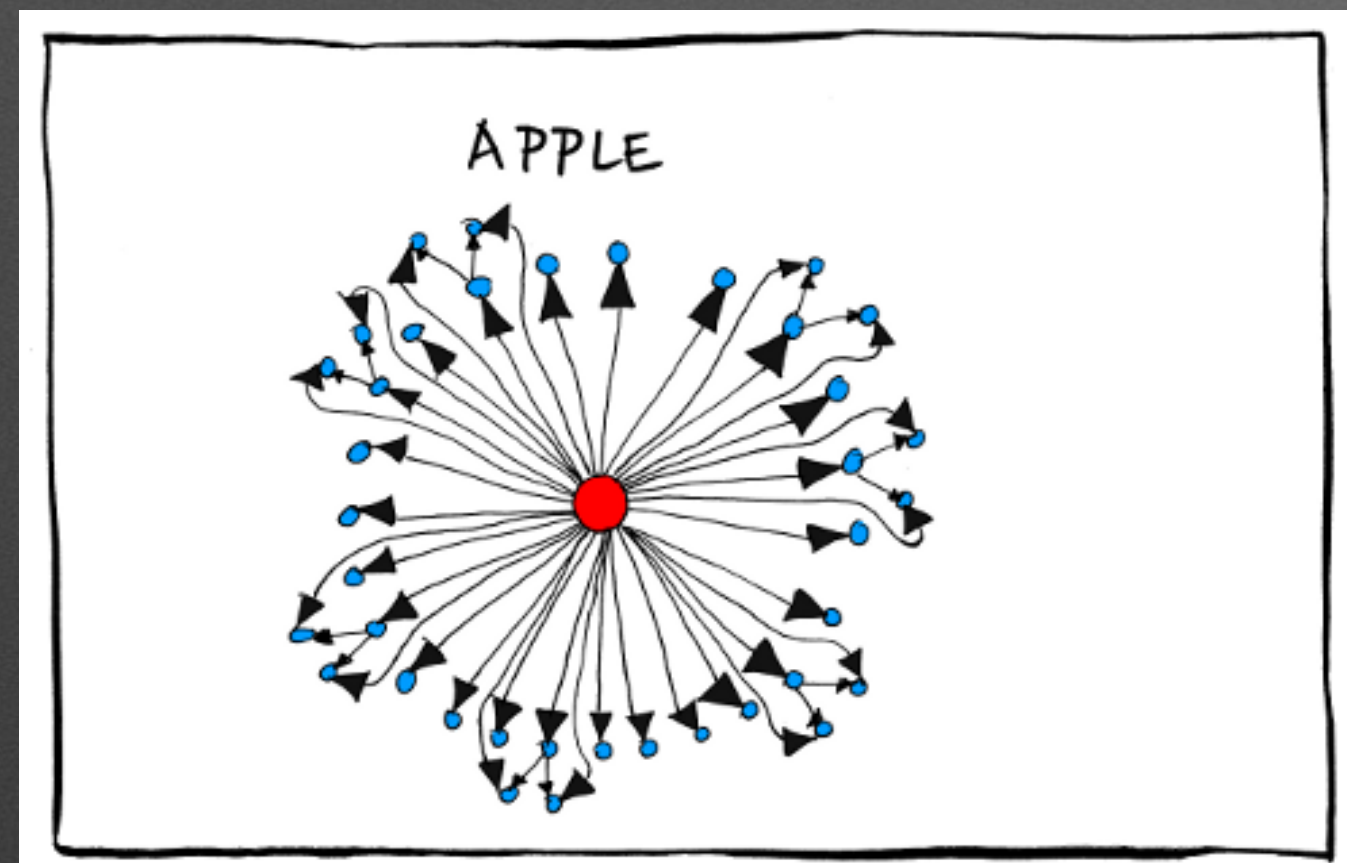


# Conway's "Law"





# Conway's Law, post-Jobs

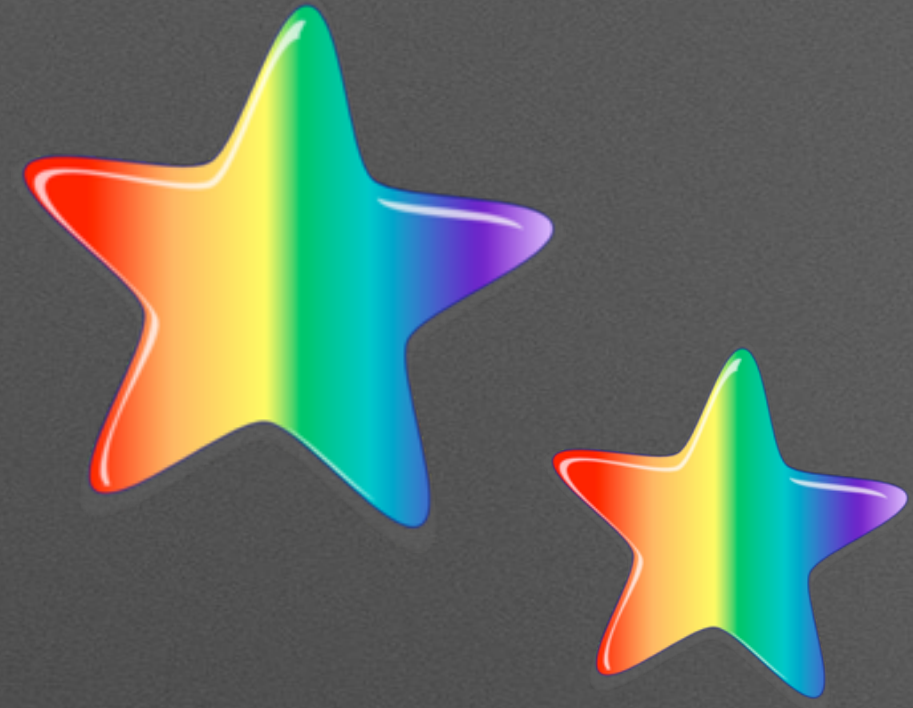




**“Conway’s Law” is not a law**







# Communication channels

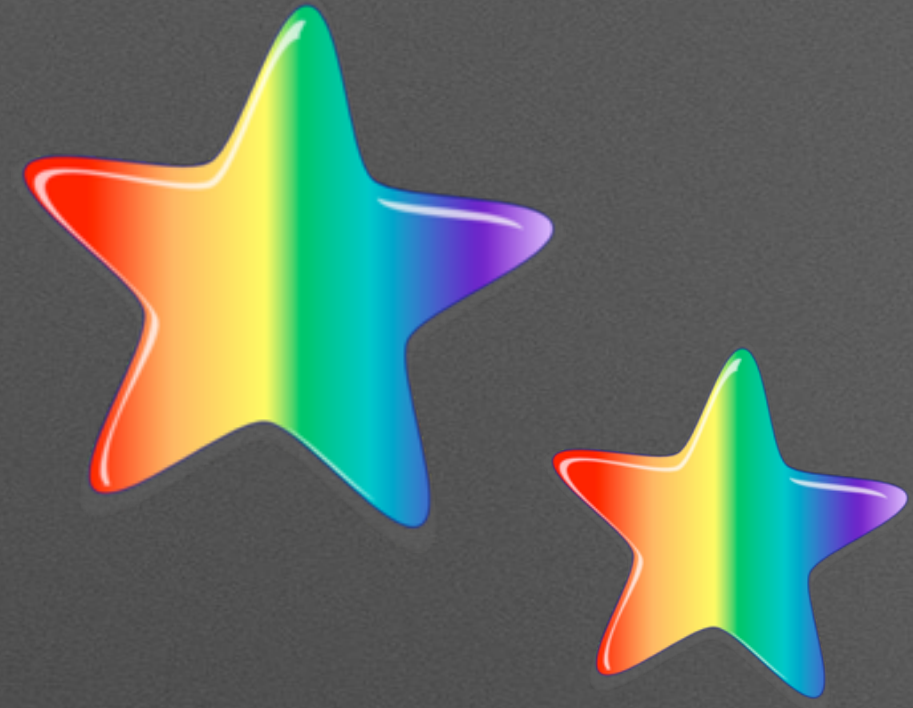
Deploys

On-Call

Pull requests, arch reviews

Observability





# Deploys





# Deploys must be:

- Fast. Rolling. Roll-back-able.
- Reliable. Breaks rarely.
- Draws a tagged vertical line in graphs.
- \*Anyone\* should be able to invoke deploy
- For bonus points: canarying or automated





**Revisit these tools regularly.**

*part of every post mortem.*



OUR DEPLOYS TOOK THREE DAYS TO RUN,  
FROM START TO FINISH.





WE NEVER MANAGED TO CONVINCING OUR PM  
THAT NOT EVERY DEPLOY NEEDED TO BE  
PERSONALLY APPROVED BY HIM.

(what the actual fuck? do it anyway.)





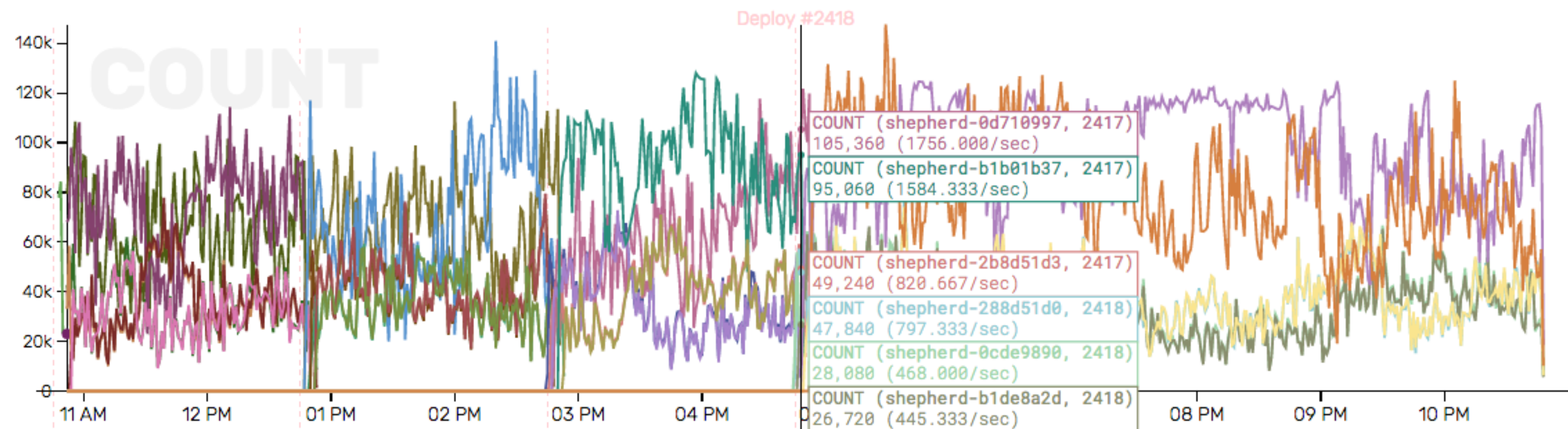
### Count and 95th Percentile of request\_dur\_ms broken down by server\_hostname and build\_id

CHANGE QUERY NAME

Run a few seconds ago

<b>TIME RANGE</b> yesterday at 10:52am to yesterday at 10:52pm  DURATION: 12 HR GRANULARITY: 1 MIN	<b>BREAK DOWN</b> server_hostname build_id	<b>CALCULATE PER GROUP</b> COUNT P95(request_dur_ms)	<b>FILTER</b> None; include all rows	<b>ORDER</b> COUNT desc	<b>LIMIT</b> None	Edit Rerun
---	--	--	---	----------------------------	----------------------	---------------

Go deeper: Enter Data Mode



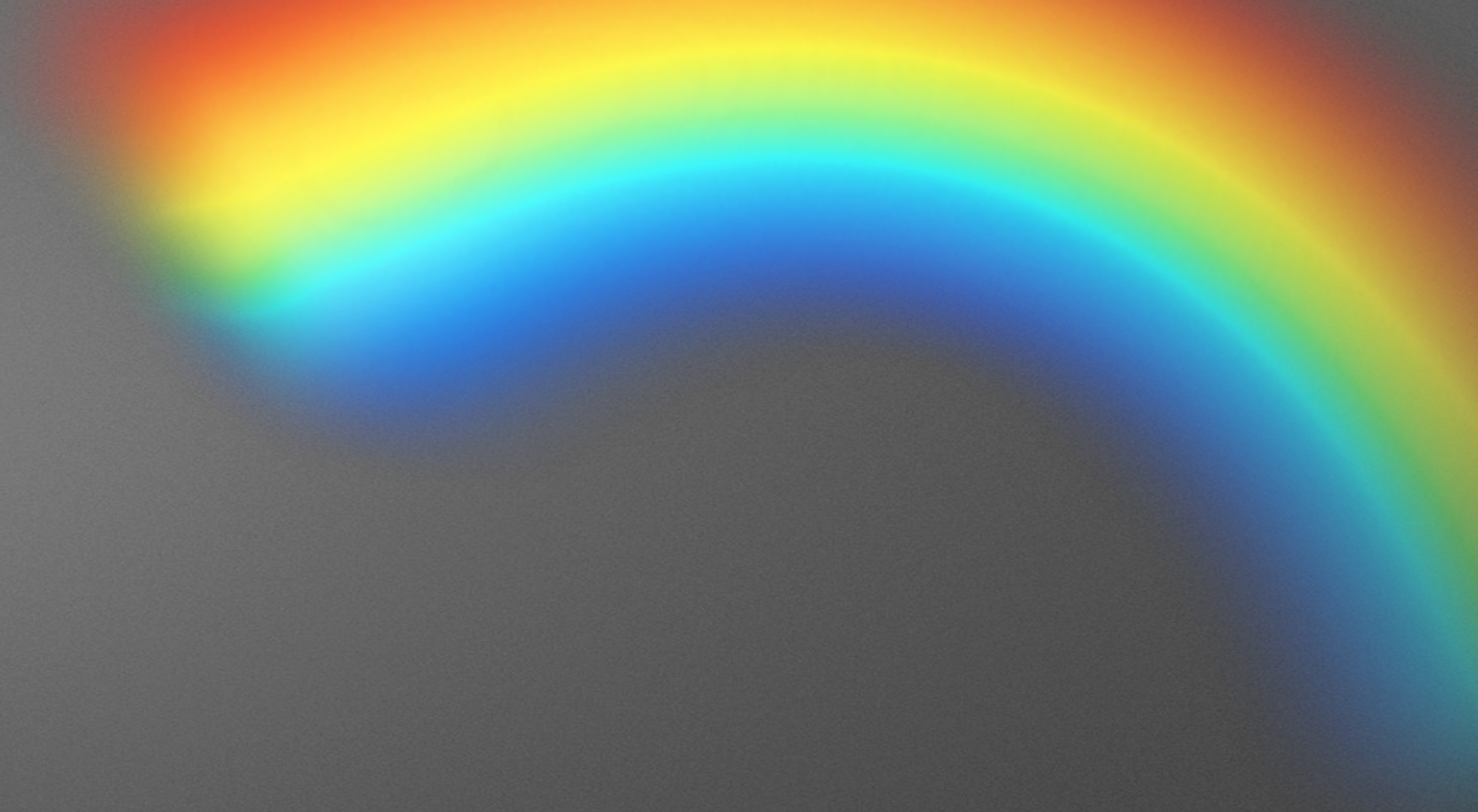
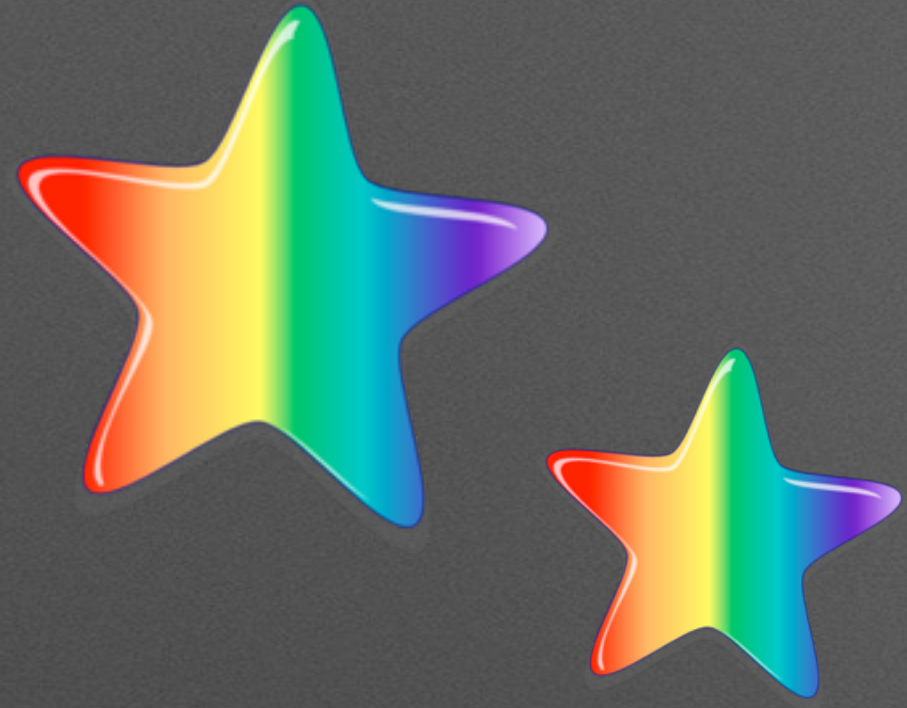
most outages are triggered by “events”,  
from humans. draw a line.



ALL SERVICES HAD TO BE DEPLOYED IN LOCKSTEP,  
AND EACH DEPLOY INVOLVED MANUALLY EDITING  
THE JSON TO SEND BACK TO MARATHON.








**On Call**





I INTERVIEWED WITH A COMPANY AS THE  
FIRST OPS HIRE. THEY SAID THEY WANTED  
ME TO COME HELP THEM LEARN HOW TO BE  
ON CALL FOR THEIR OWN SERVICES.

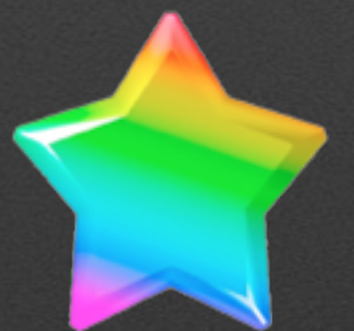
I JOINED, AND THEN LEARNED THEY MEANT  
"SOMEDAY, MAYBE YEARS FROM NOW."



# On call questions



- Who is on call? Is it a necessary part of being an engineer?
- How many rotations are there?
- How often do people get woken up? \*who\* gets woken up?
- How do you know? Who keeps track?
- Are there different rotations for stateful and stateless services, front-end and backend?
- Is there an escalation path?





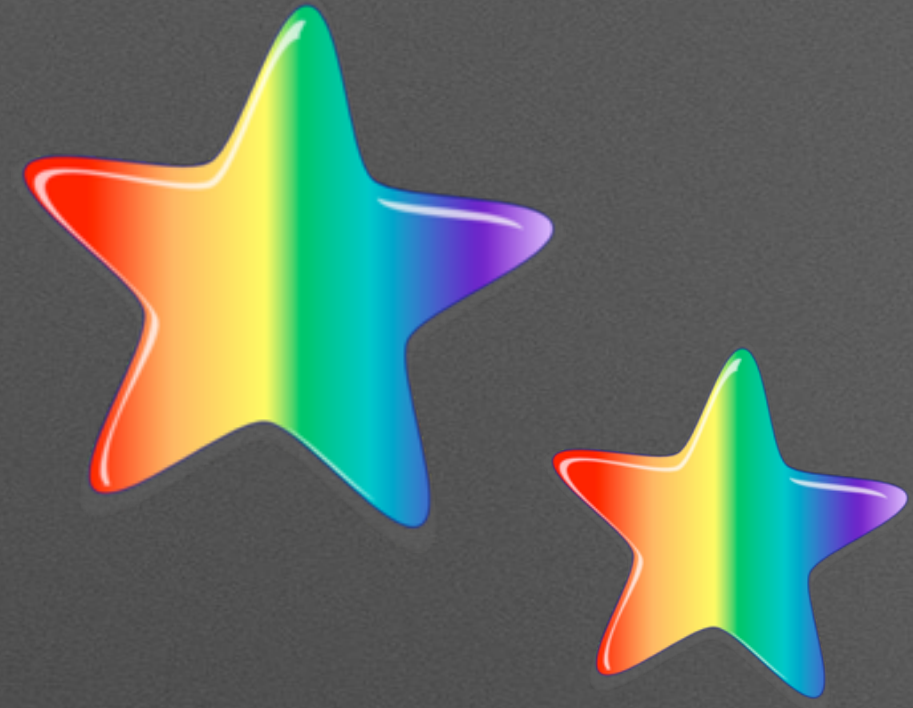


**change is the only constant**

**seek feedback**

**move forward <3**





# What should leaders know?

Managers, tech leads, and engineers





# smart nodes, dumb pipes

Managers' job is primarily facilitating nodes



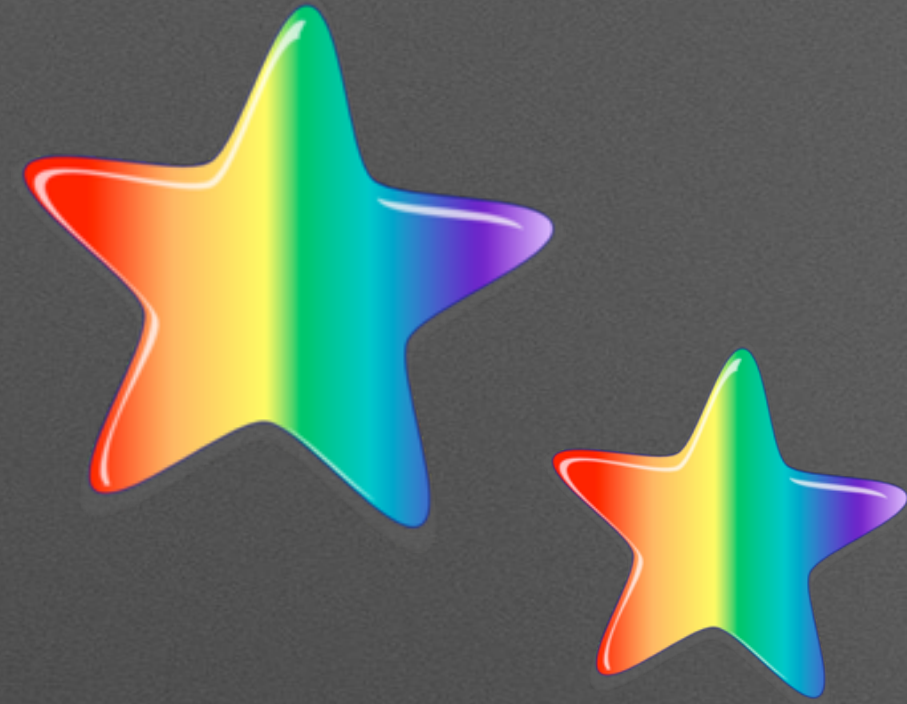




# Things about leadership

- Leadership is not a zero sum game. The best leaders try to empower literally everyone to perform a leadership role in at least some areas.
- Create guard-rails, not walls.
- Be conventional in the big things (salary, org), unconventional in the small.
- If you give a shit about diversity, don't wait til you're "ready" to hire them ... look for ways to support underrepresented groups now. Make friends. Help people. Diversify your friend groups and personal networks. Be creative.





# Management

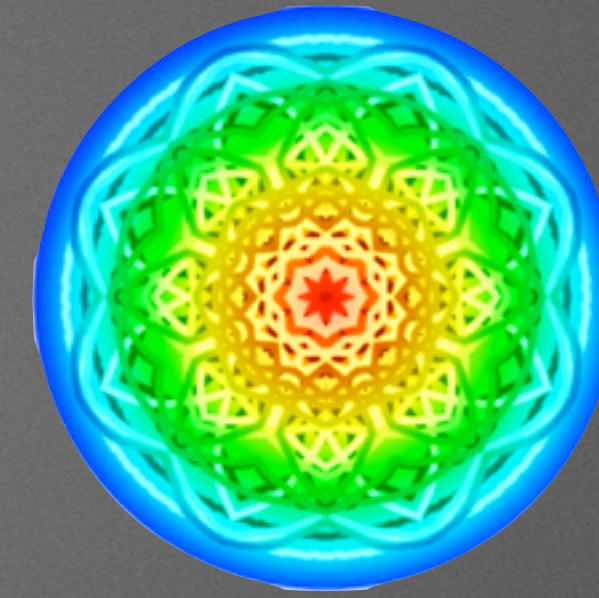
- Put the humans first, and the mission a close second
- Be an enabler. Don't starve your tech leads of growth opportunities by sucking all oxygen.
- Reward intentionally.
- Leadership is not zero-sum, encourage leadership everywhere
- Managers, be friends with each other! Tolerance is not enough



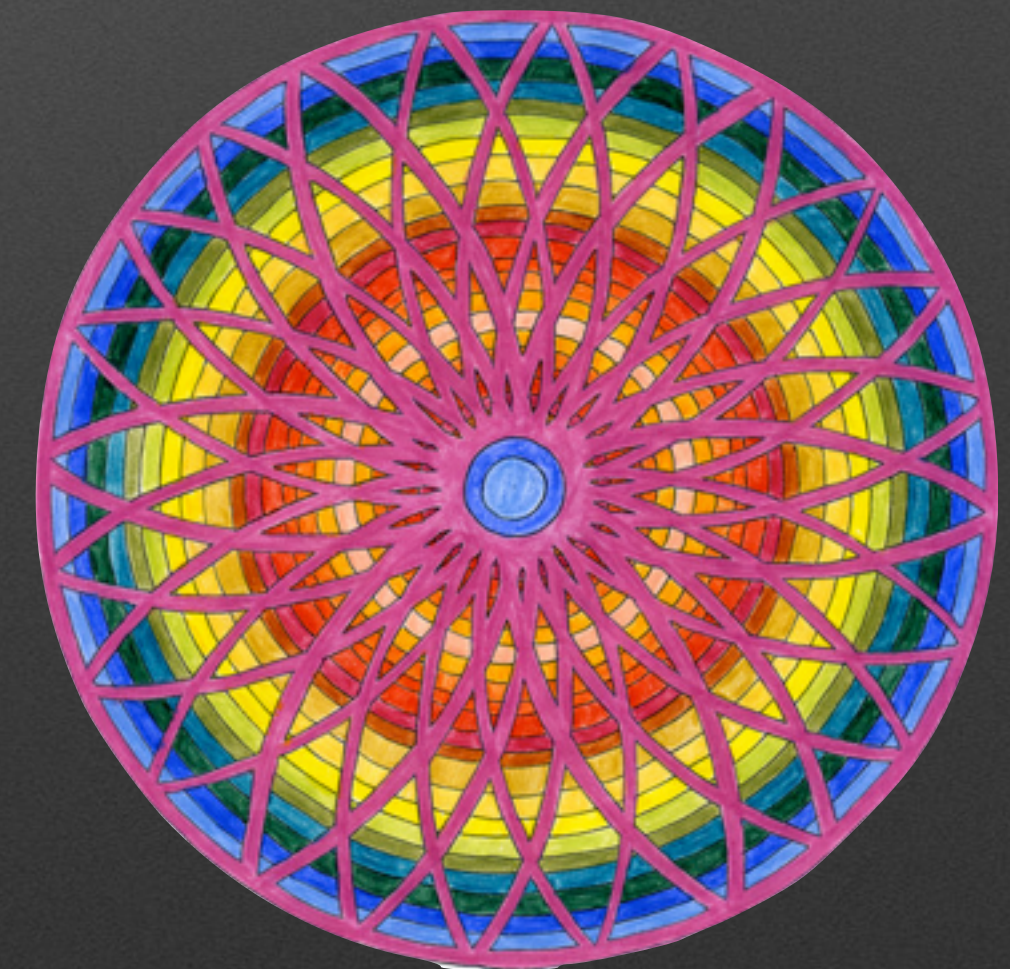
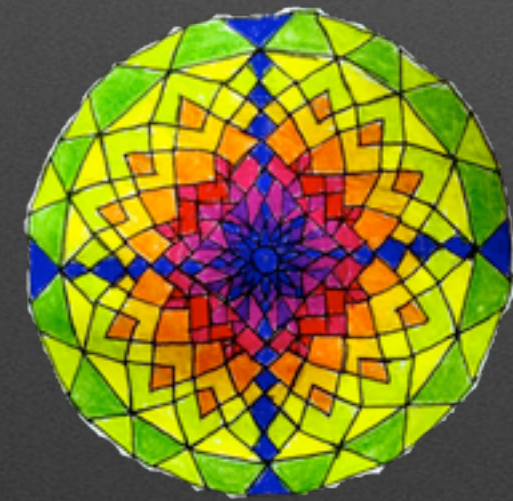
**The most powerful weapon in your arsenal  
is always cause and effect.**







**Engineers should be on call  
for their own services.**







# Corollary: on-call **must** not be hell.

- Guard your people's time and sleep
- No hero complexes. No martyrs.
- Don't over-page. Align engineering pain with customer pain
- Roll up non-urgent alerts for daytime hours
- Your most valuable paging alerts are end-to-end checks on critical code paths.





**Probe every software engineering candidate  
for their ops experience & attitude.**



... yep, even FE/mobile devs!



you are signaling ...



**“Operations is valued here.”**







Operations engineering is about making systems maintainable, reliable, and comprehensible.

Senior software engineers should be reasonably good at these things.

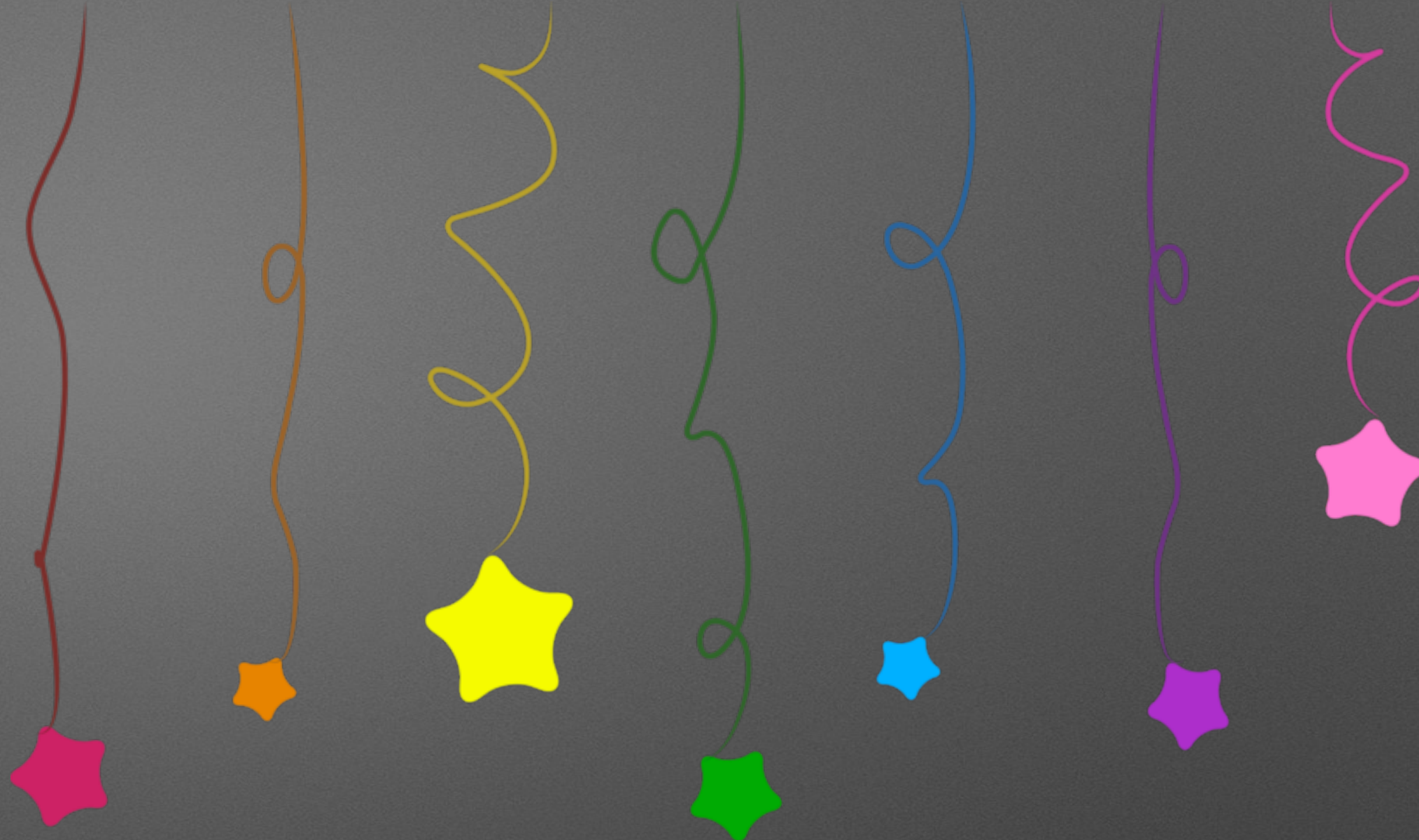
**So if they are not, don't promote them.**





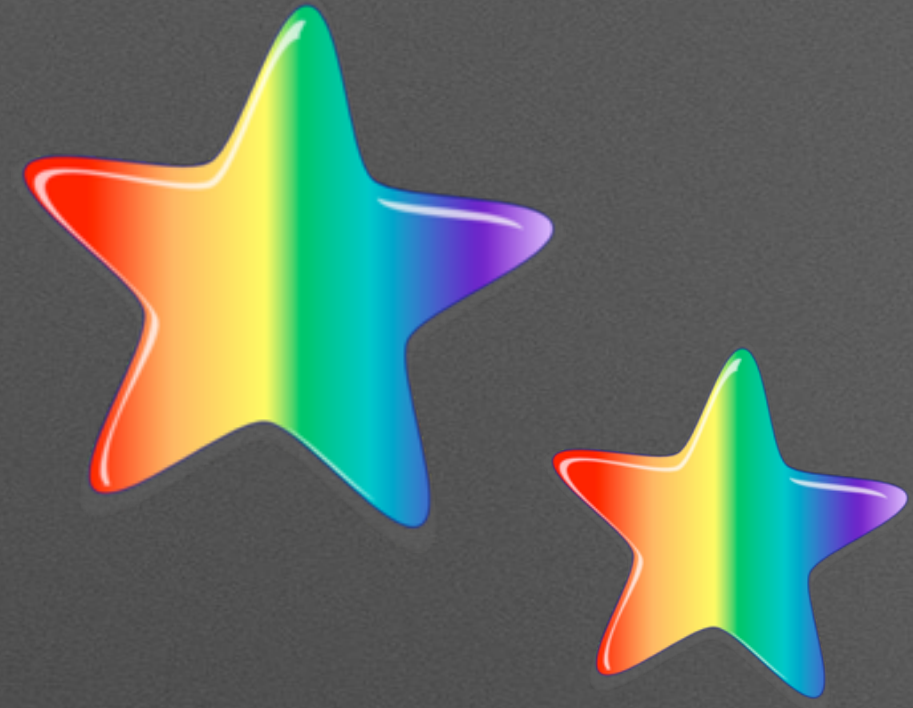
**Choose the problems you are not  
going to solve, or they will choose you.**





**Get buy-in from *\*all\** stakeholders.**





**Tech leads, senior ICs**





# Making decisions:

Get ready to talk to people a lot more about microservices.  
Sorry!





# accountability: still a bitch

WHAT \*REALLY\* PISSES ME OFF IS THAT NOW, INTERNAL-TO-THE-COMPANY, ANY TECH THAT WAS USED AS PART OF THAT ILL-FATED CLUSTERFUCK

SO ANY SORT OF CONTAINER TECHNOLOGY IS MET WITH HELLA RESISTANCE, EVEN THOUGH THE CONTAINERS WE WERE USING WEREN'T THE SOURCE OF ANY OF THE ISSUES.



#truestory





# Yes but ....

Yes, microservices helps you drift a little bit and innovate independently ...

BUT, not as much as you might think.

You all still share a fabric, after all.

Stateful still gonna ruin your party. (and IPC, sec discovery, caching, cd pipelines, databases etc.)



ONE MANAGER MANAGED TO CAPTURE ALL SERVICE DEVELOPMENT IN THE ORG. SO ANY NEW WEB SERVICE HAD TO BE DONE BY HIS TEAM (ABOUT 10% OF THE TOTAL FTES). MOST OF WHOM COULD ONLY DRAG-AND-DROP IN THE WPS GUI.

SO WE START STANDING UP REST MICROSERVICES. AND IMMEDIATELY HE'S ANGLING TO GET HIMSELF IN CHARGE OF THE TEAM MANAGING THE GATEWAY IMPLEMENTATION, ARGUING HE NEEDS TO CENTRALISE THE DEVELOPMENT OF THE MICROSERVICES IN ONE TEAM TOO...

#truestory





AFAICT, IT CAME OUT OF A PISSING CONTEST THE VPENG  
WAS HAVING WITH THE VP OF IT OVER WHO, I DUNNO,  
HAD THE BIGGEST BALLS OR SOMETHING.

THEY BOTH LAUNCHED HUGE, WRONG-HEADED PROJECTS AT THE  
SAME TIME AND THEN FOUGHT OVER RESOURCES.

NEITHER VP HAD ANY USE FOR PEOPLE THAT WEREN'T YES-MEN.



- I don't think anyone should approach management as a thing they move in to permanently. It's psychologically disfiguring.
- Nor is the maturation process one way. New teams within the company should be springing up. Hackathons can be a great way, esp if it involves dogfooding. Empathy needs constant renewal.
- Practice making mistakes together. Practice cheerful apologies, asking questions, giving awkward feedback. It gets easier.





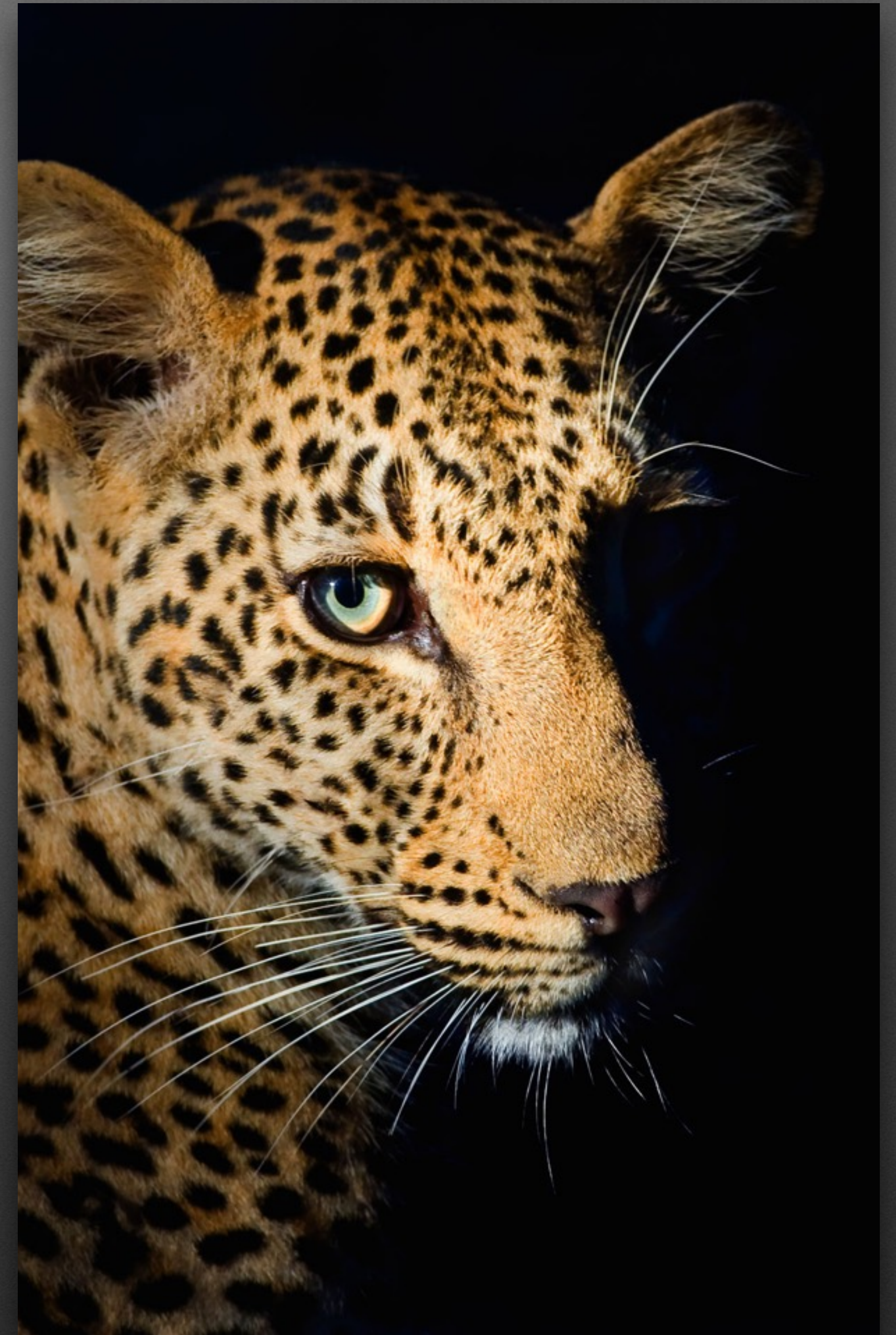
I CAN'T POSSIBLY BE ON CALL FOR MY OWN SERVICES.  
IT'S TOO NOISY ... I WON'T GET ANY WORK DONE!

MY SOFTWARE ENGINEERS' TIME IS TOO VALUABLE  
TO SPEND ON RELIABILITY WORK. WE NEED  
THESE FEATURES TO SHIP.



# Unit tests for your org

- What is your mission as an org? Does everyone have a similar answer? (No, they don't.)
- One-on-ones. With your reports, your peers, your skip levels up and down, with key partners elsewhere. No replacement.
- Ask the same questions periodically of everyone in your team. Ask a creative, brand new question once a week.
- Ask your team how you will know if they are unhappy, bored, or frustrated. Watch for those things.
- Sit there quietly so they have to ask you questions.
- Sit there in silence until they answer things, don't fill in the answers.
- Look for the uncomfortable places. Be happy when you find them.





# **There is no fairy-tale answer**

Leadership means engaging creatively with the process  
and constant experimentation and change.





April 2012 @ VMware

# Charity Majors @mipsytippsy

