# why are DISTRIBUTED SYSTEMS so hard?

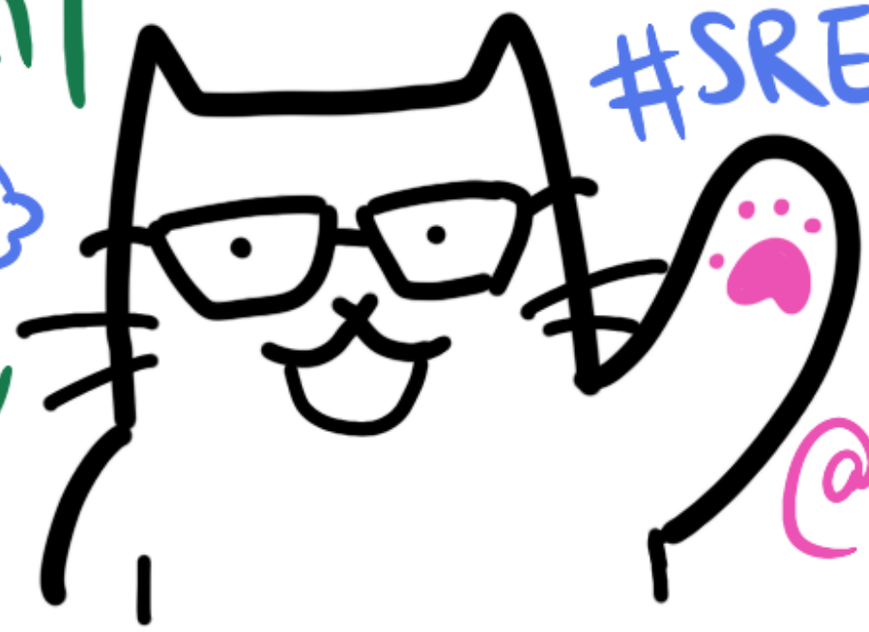
@deniseyu21

deniseyu.io

Software Eng @ Pivotal Cloud Foundry

Toronto-based

Hello, #SREcon!



@deniseyu21

Tech-doodling enthusiast

deniseyu.io/art



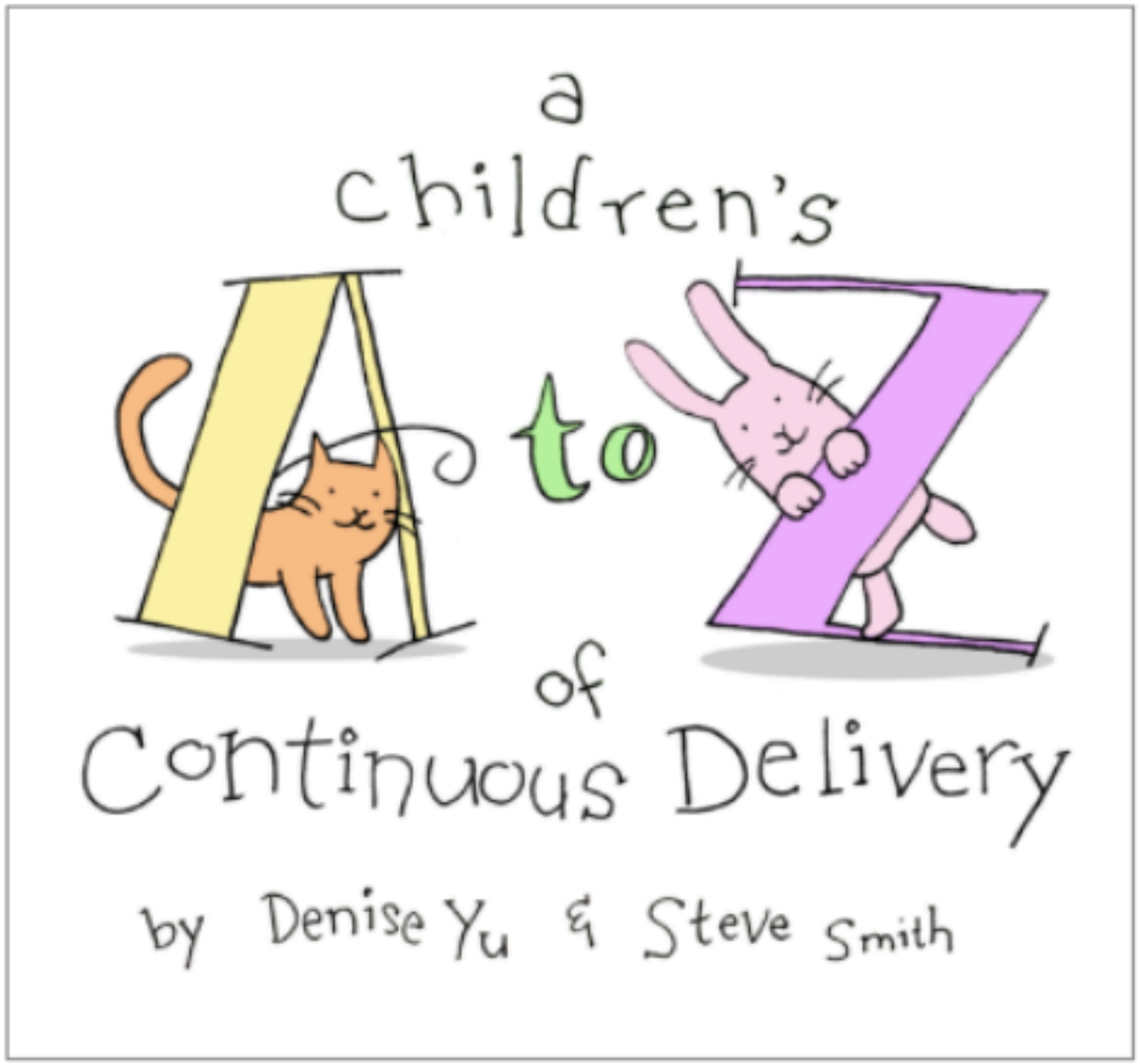


deniseyu.io/art

# A Children's A to Z of Continuous Delivery

Denise Yu and Steve Smith

A friendly introduction to key concepts in Continuous Delivery, for all ages




leanpub.com/achildrensatozofcontinuousdelivery

# AGENDA

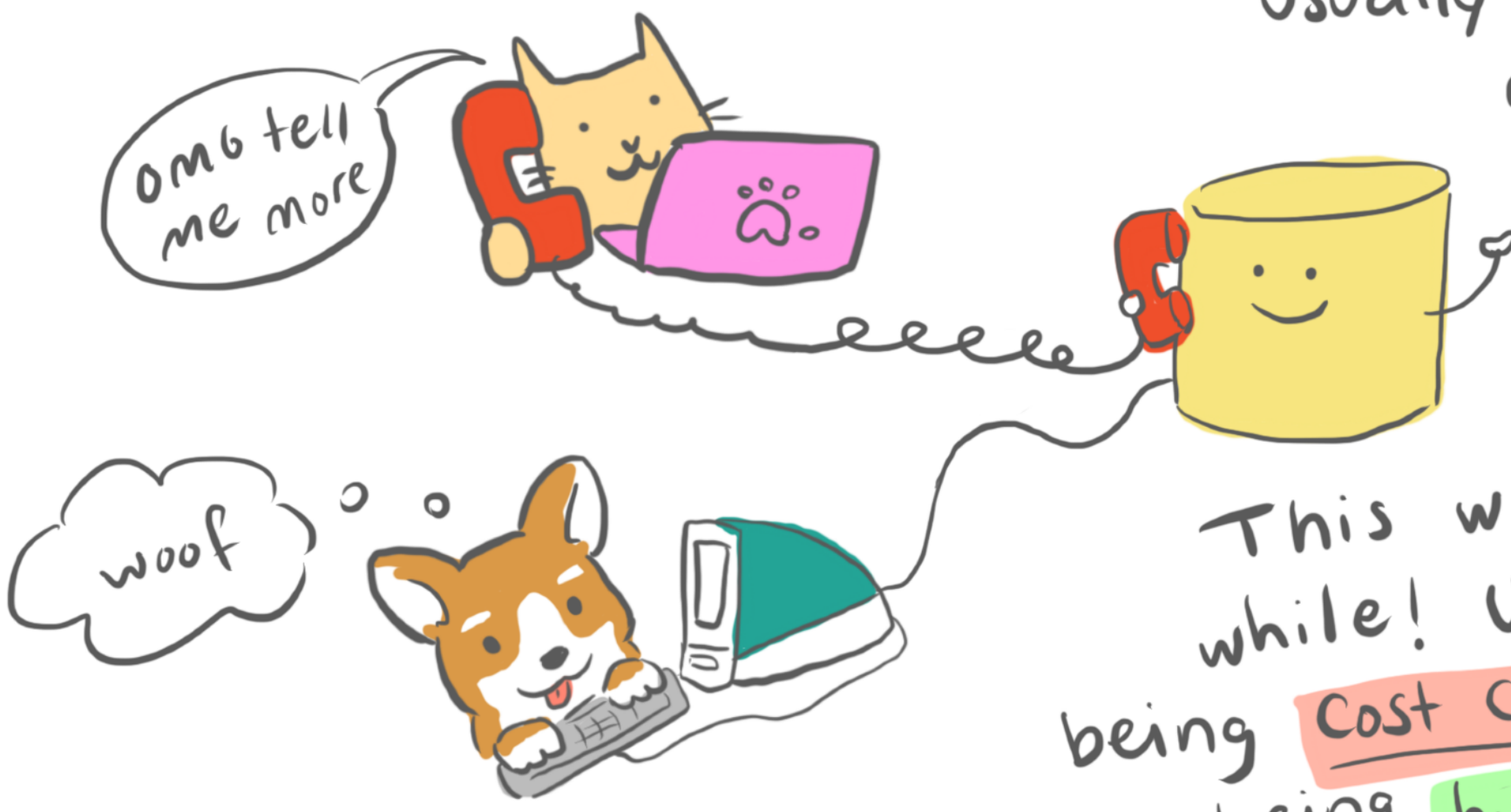
- why is distsys a thing
- re CAP
- networks are hard
- how to make life better?


@deniseyu21

A long time ago,

in a datacenter not too far away...

All business applications talked to one database, usually hosted on a company's own hardware.




This worked for a while! Until IT stopped being Cost centers & started being business enablers.

@deniseyu21

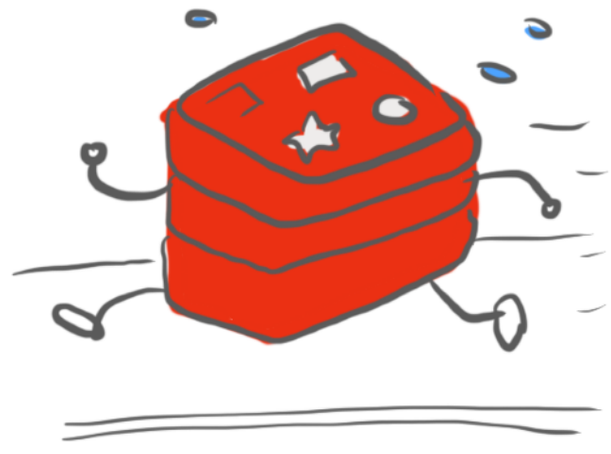
Data storage & retrieval needs evolved as software became business differentiators.

Business analysis & data warehouses

ML & Natural Language processing

Faster, bigger queries!





@deniseyu21

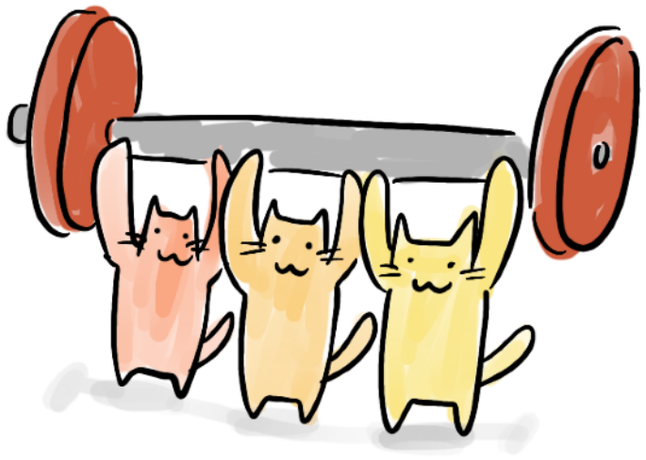
So we scaled vertically...


until unit economics (or physics) caught up.

@deniseyu21


@deniseyu21

# REASONS TO HORIZONTALLY DISTRIBUTE:



Scalability:

one machine cannot handle request or data size

Availability:

if one machine goes down, others keep working


Latency:

go faster when data is stored geographically closer to users

@deniseyu21

You may have heard the term "shared nothing" architecture:


Machines do not share access to any resources.

(This makes things Complicated quickly.)

@deniseyu21

what does it actually mean to run a distributed system?

Many machines,
running many processes,
and "only message-passing
via unreliable networks
with variable delays,
and the system may
suffer from partial failures..."

Martin Kleppmann,
"Designing Data Intensive Applications"

?

@deniseyu21

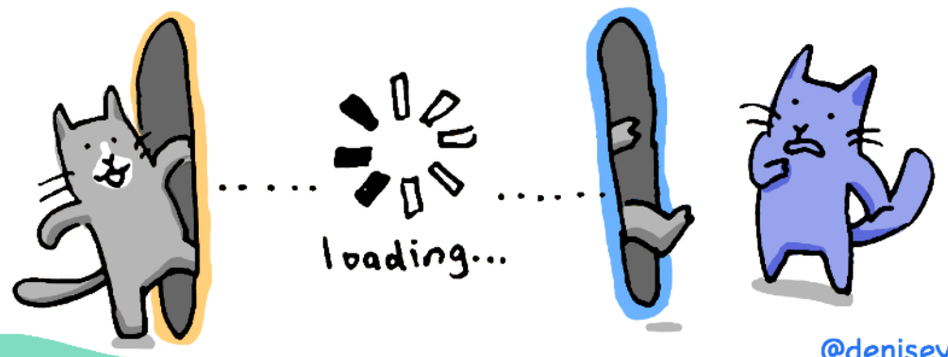
# the 8 Fallacies of Distributed Computing

@deniseyu21

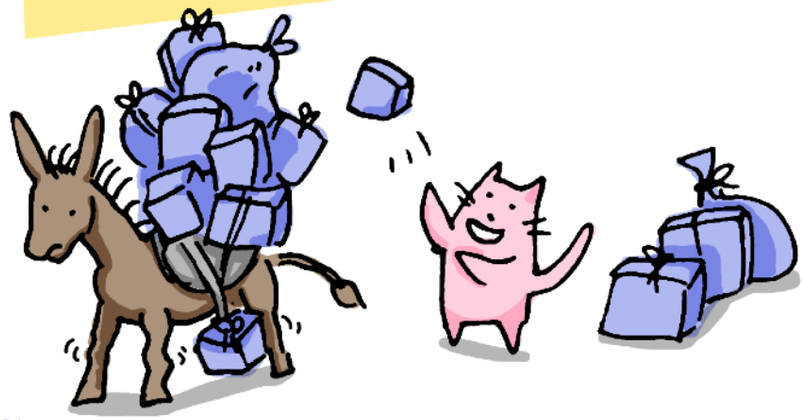
Originally formulated by L. Peter Deutsch & Colleagues at Sun Microsystems in 1994; #8 added in 1997 by James Gosling

## 1) The network is reliable

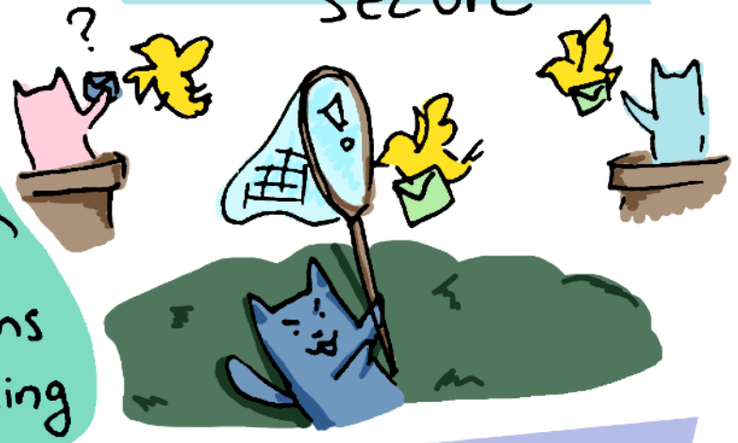

## 2) Latency is ZERO



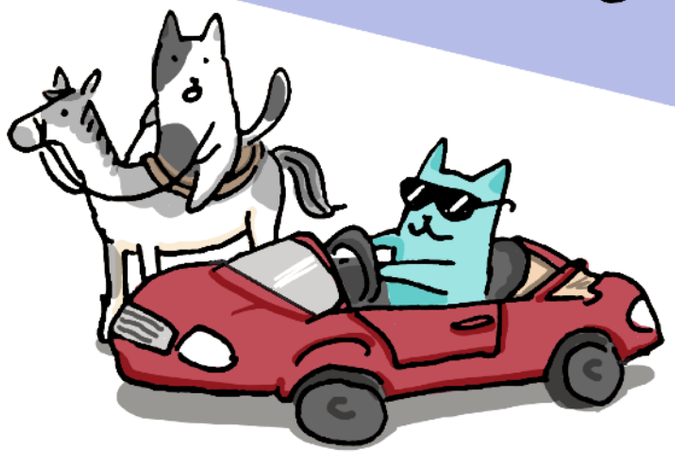

## 3) Bandwidth is infinite



## 4) The network is Secure




## 5) Topology doesn't change



## 6) There is only one administrator


## 7) Transport costs $0

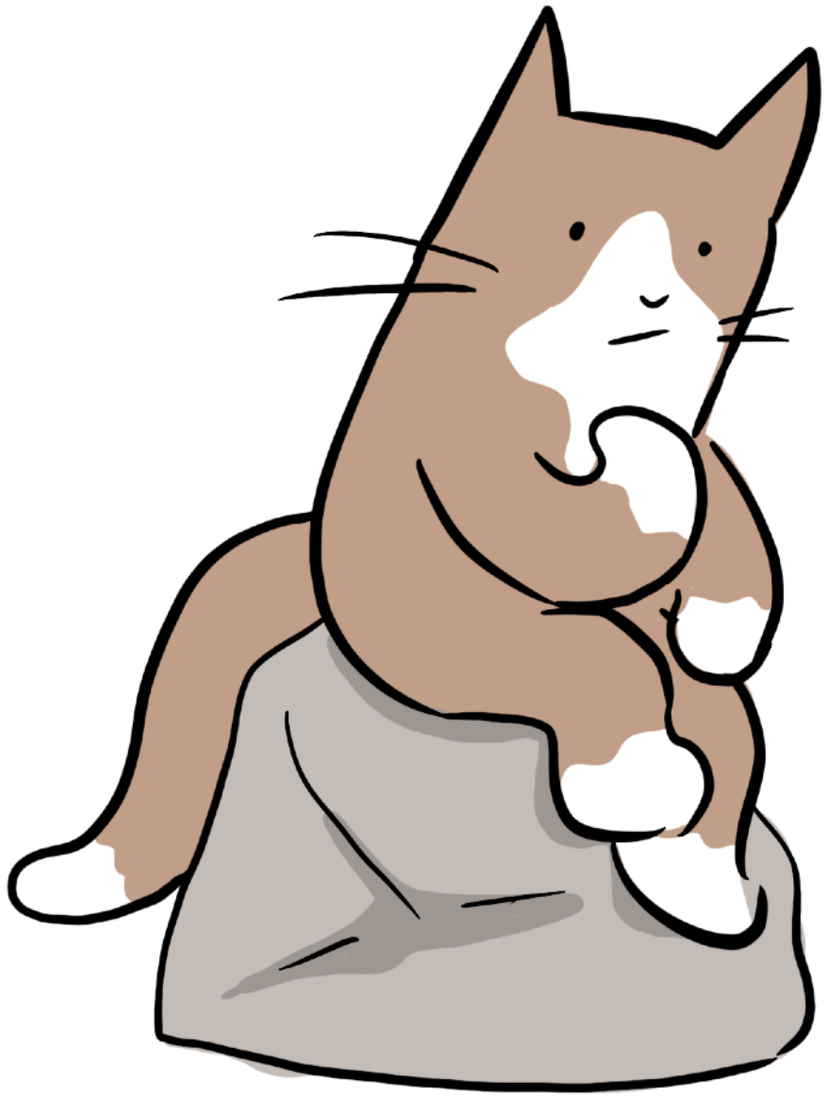

## 8) The network is homogeneous

# 1 The network is reliable

So much unreliability!

How can we even know what is true about the state of the world?

@deniseyu21



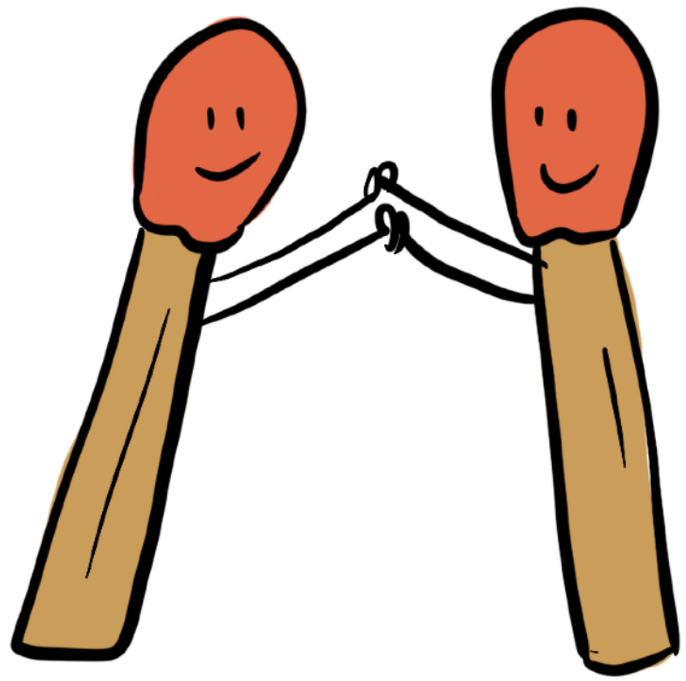
This is an epistomology problem!

@deniseyu21

there are
TWO SCHOOLS
of epistemic PHILOSOPHY

FOUNDATIONALISM

Fundamental truths
like math first principles

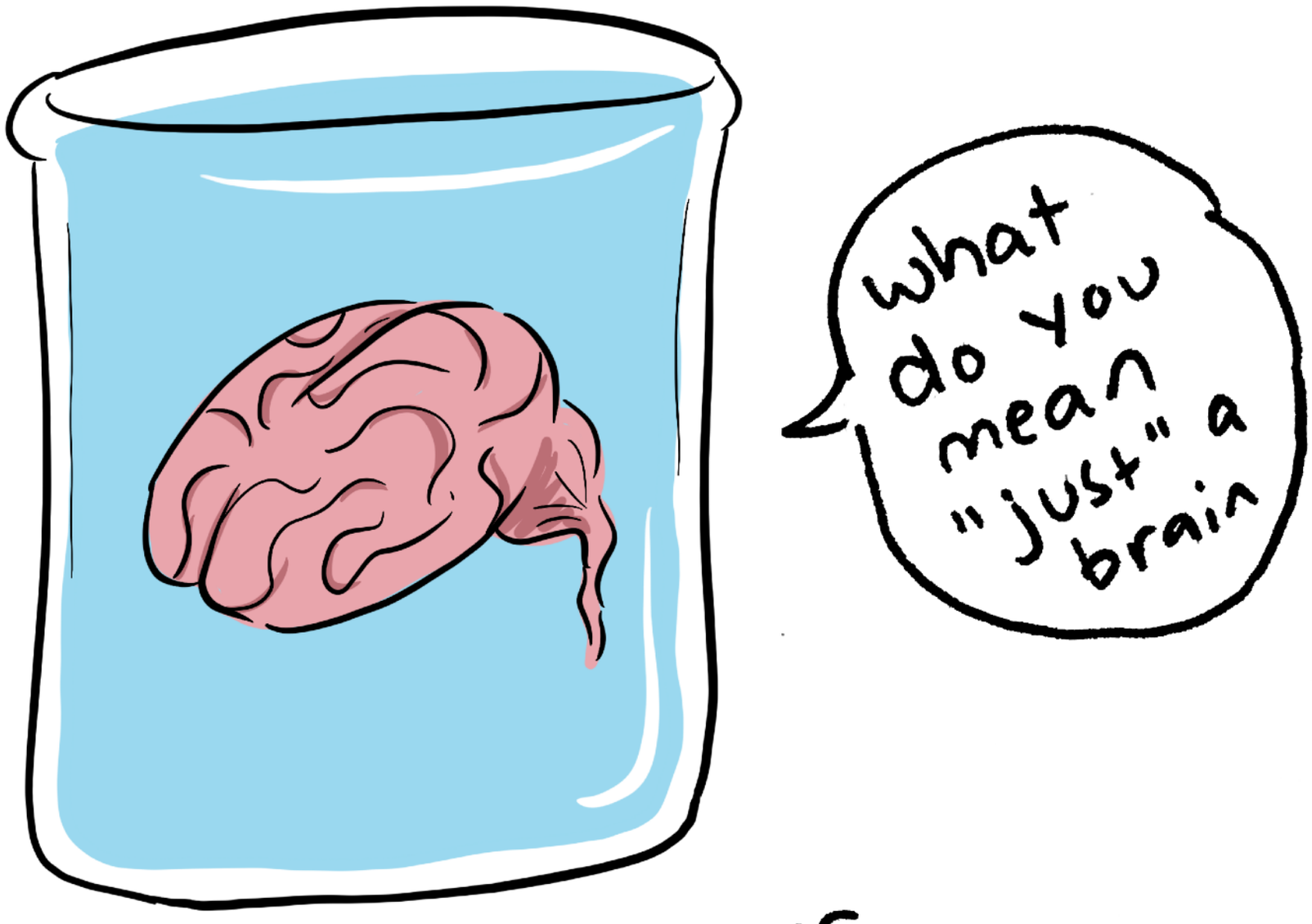


COHERENTISM

Logical, interlocking,
mutually-reinforcing
truths like
matchsticks

@deniseyu21

AND WHAT IF WE'RE ALL

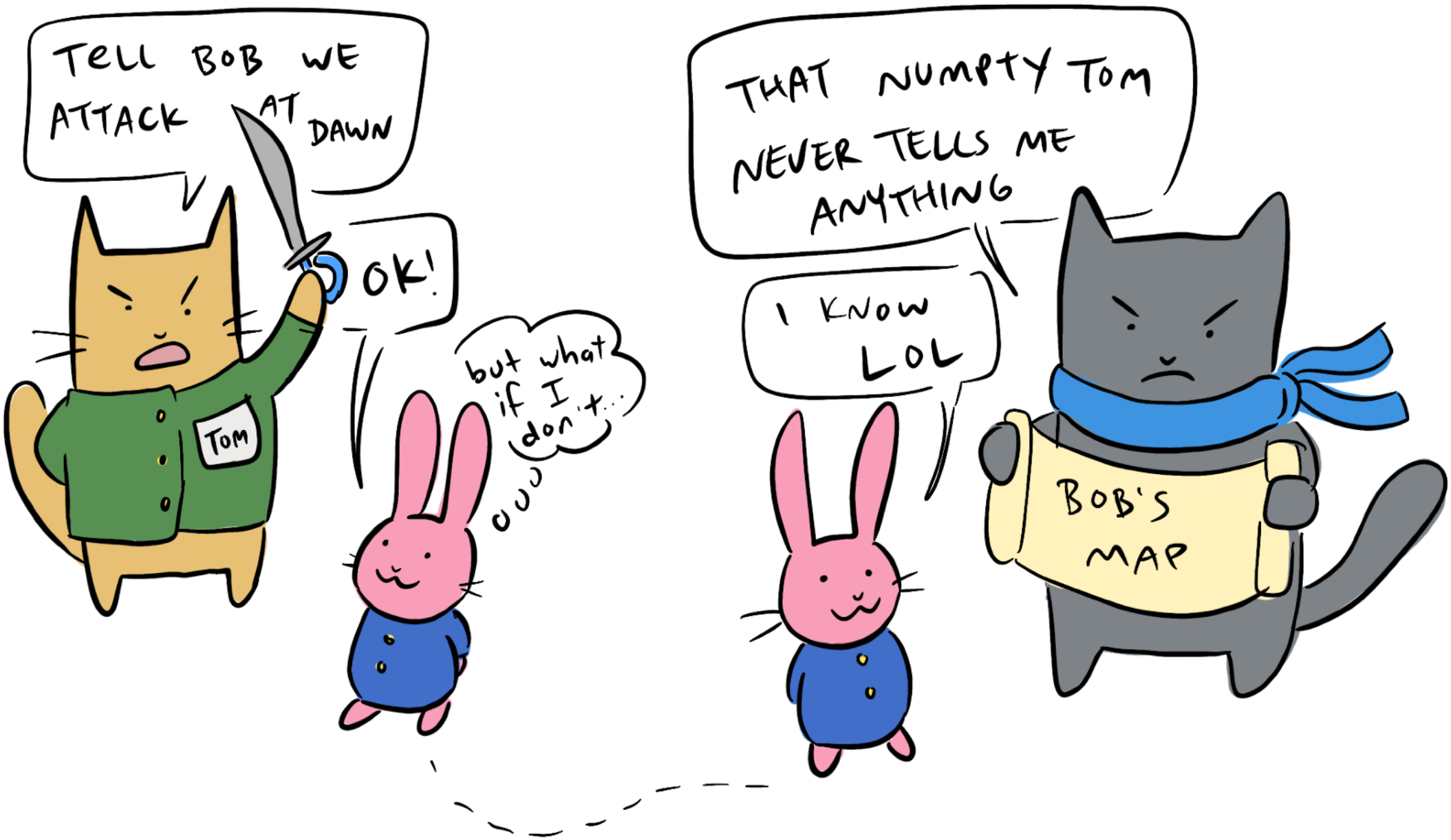



JUST BRAINS IN VATS

#skeptics

@deniseyu21

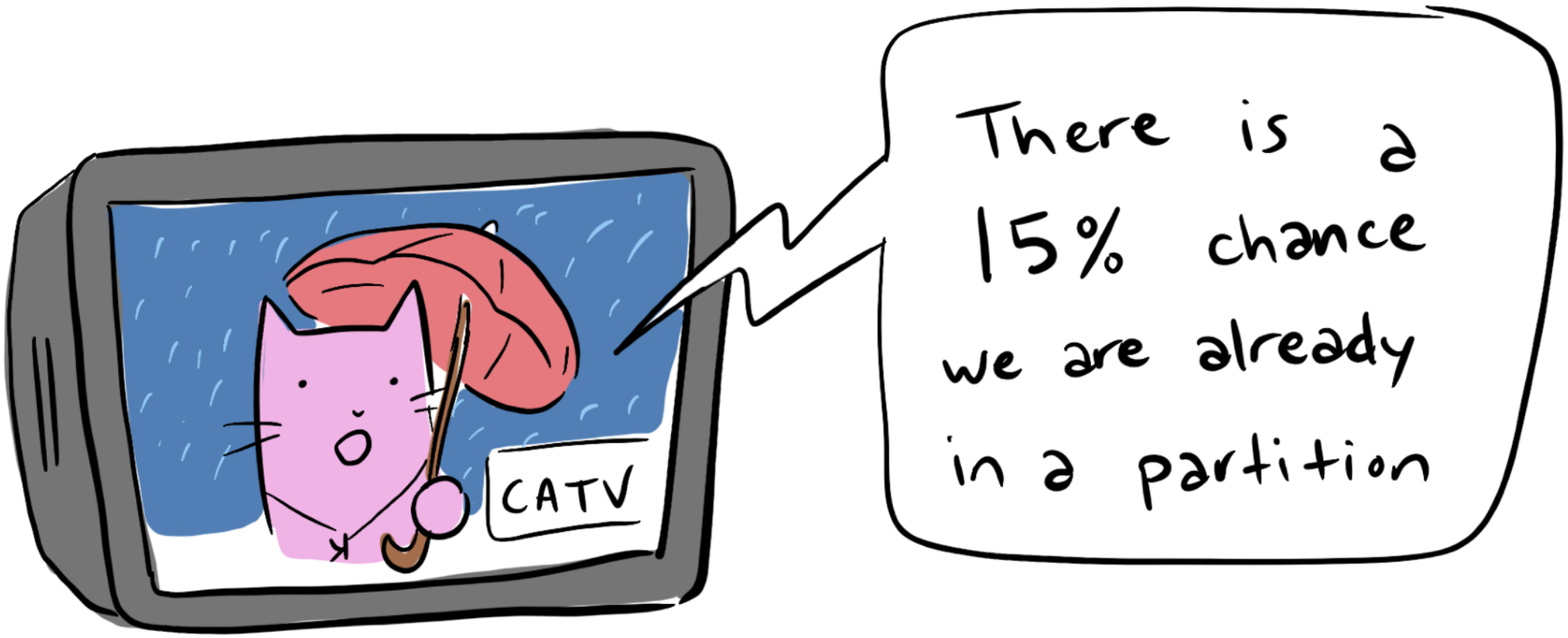
# The Byzantine Generals Problem



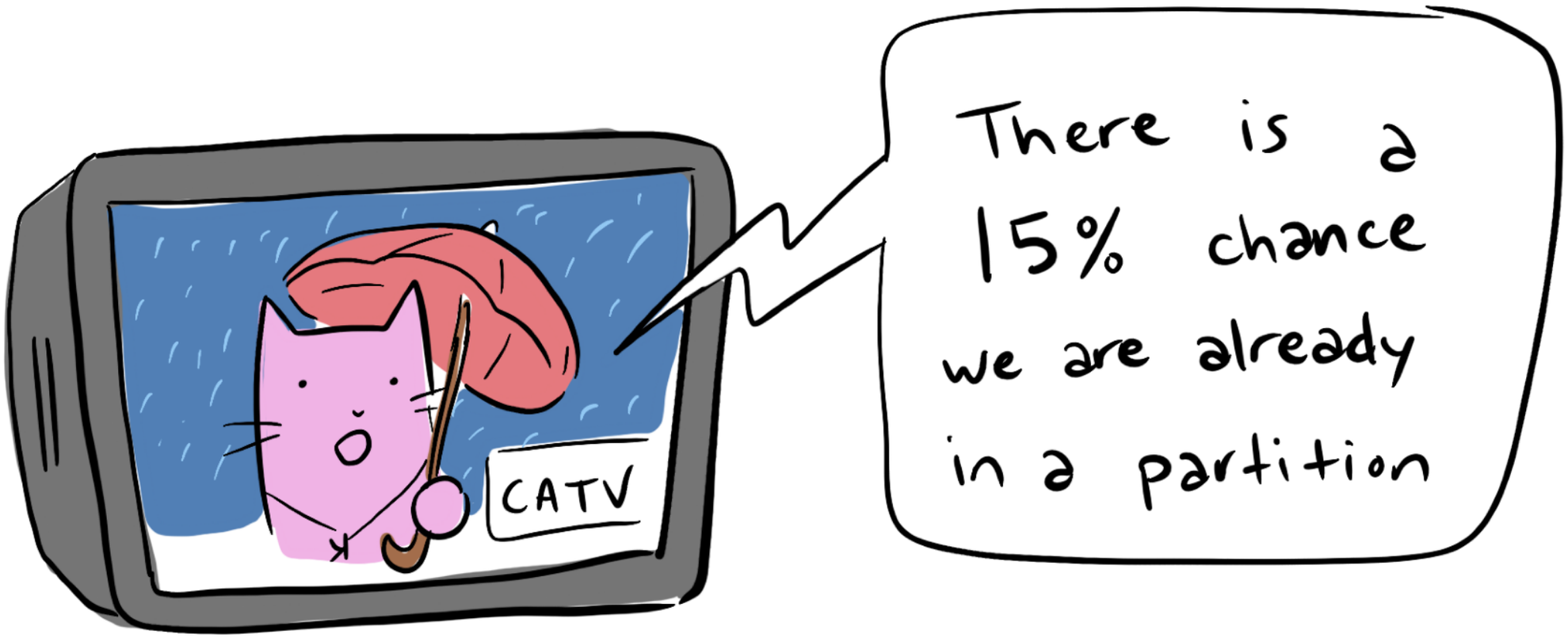

@deniseyu21

# MONITOR & observe

@deniseyu21




There are lots we can't know. But in distributed computing, we can know one thing:




There are lots we can't know. But in distributed computing, we can know one thing:

Shit's gonna fail

@deniseyu21

CAT

THEOREM

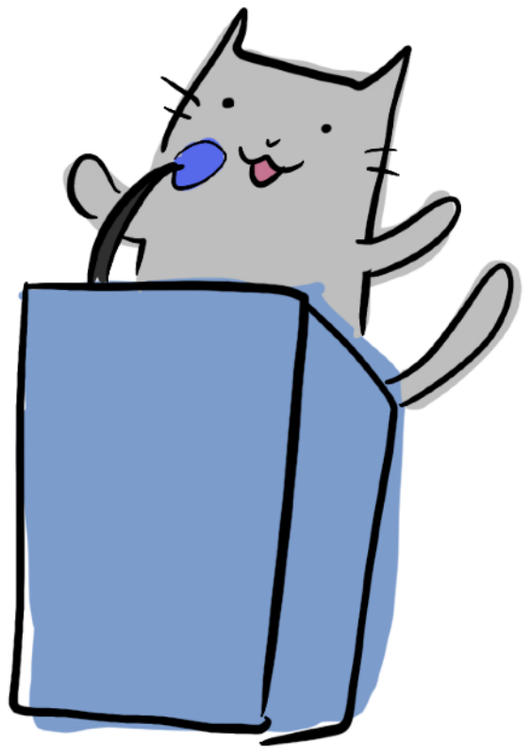
@deniseyu21

2000

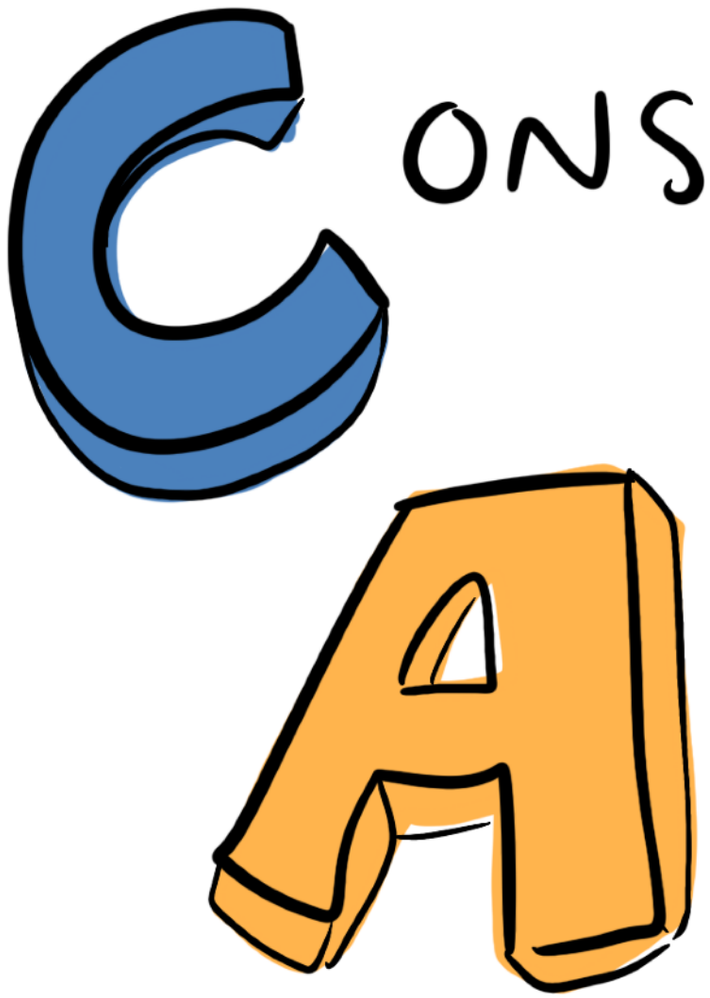
DR ERIC BREWER

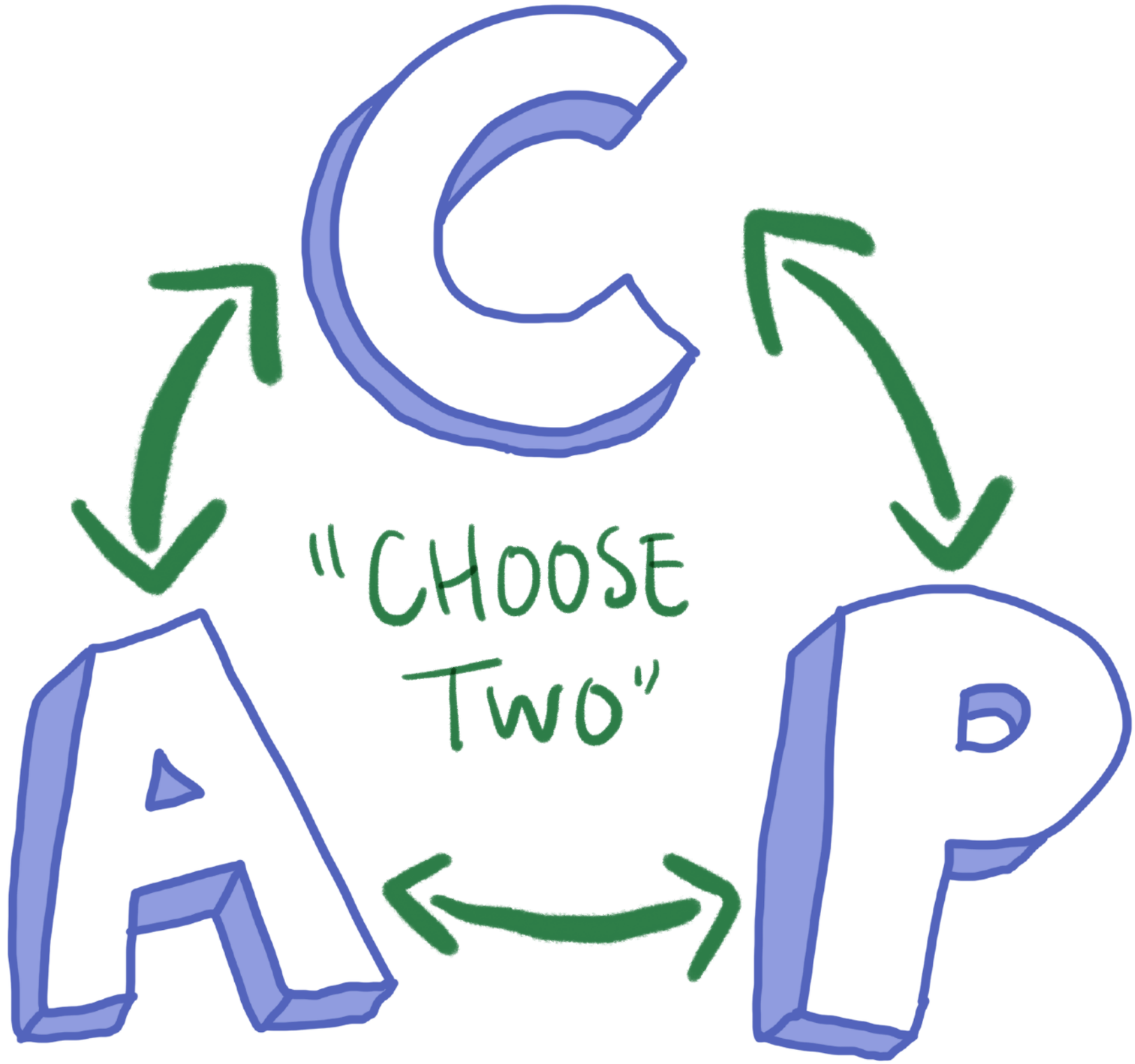
"Towards Robust Distributed Systems"

@ Principles of Computing Conf

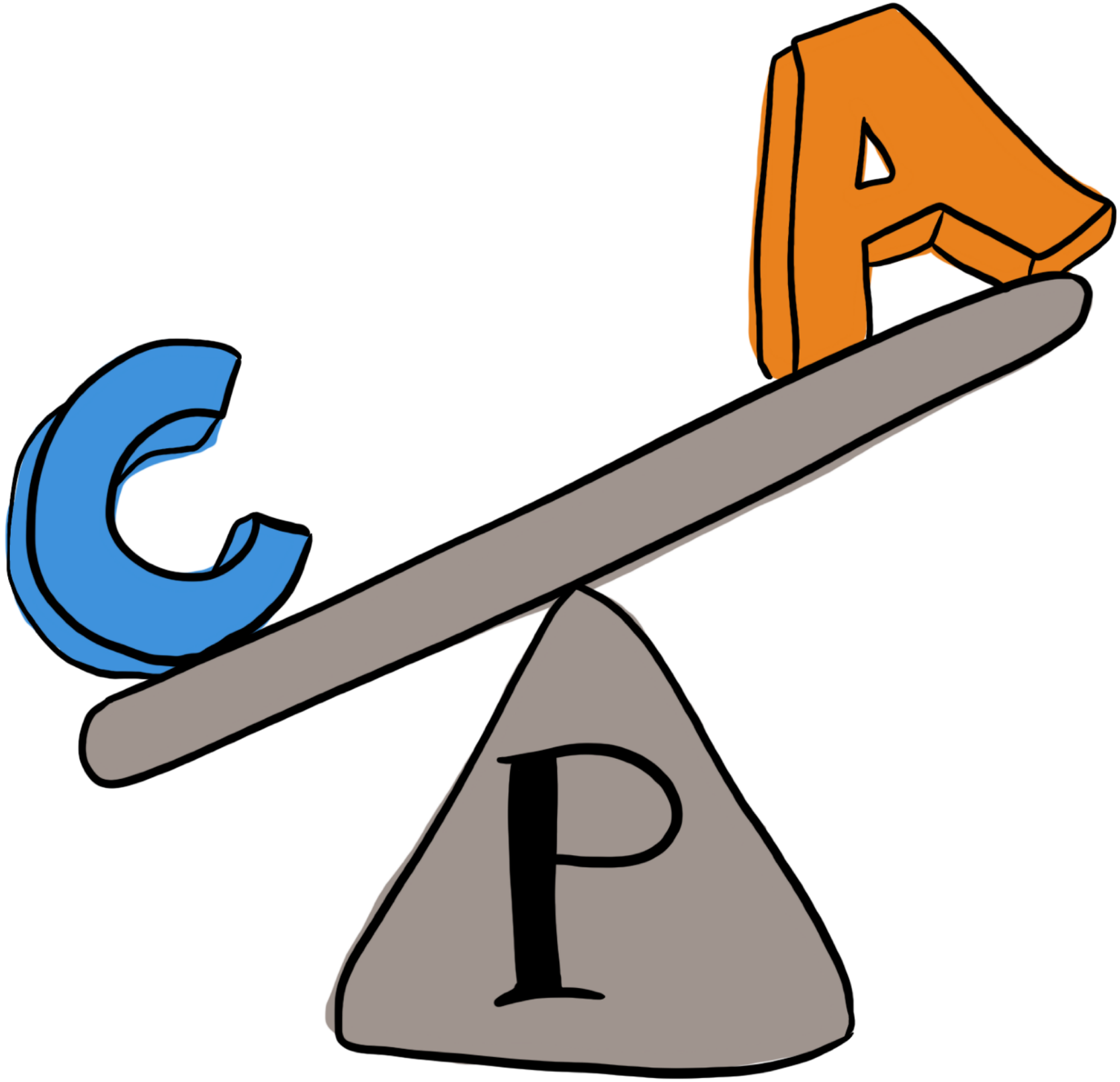


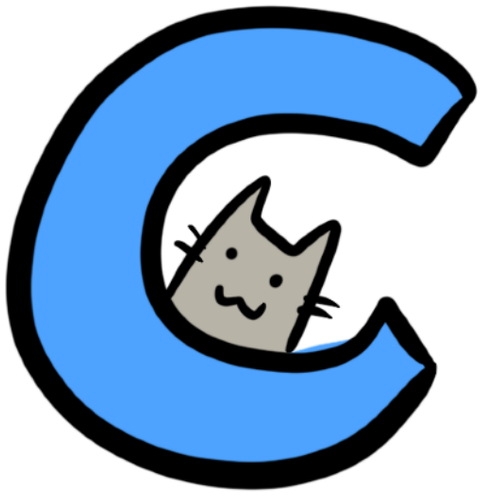
@deniseyu21


Consistency *
Availability
Partition tolerance


C
"CHOOSE
TWO"
A
P


NO!


C
A
P



# IS FOR LINEARIZABILITY

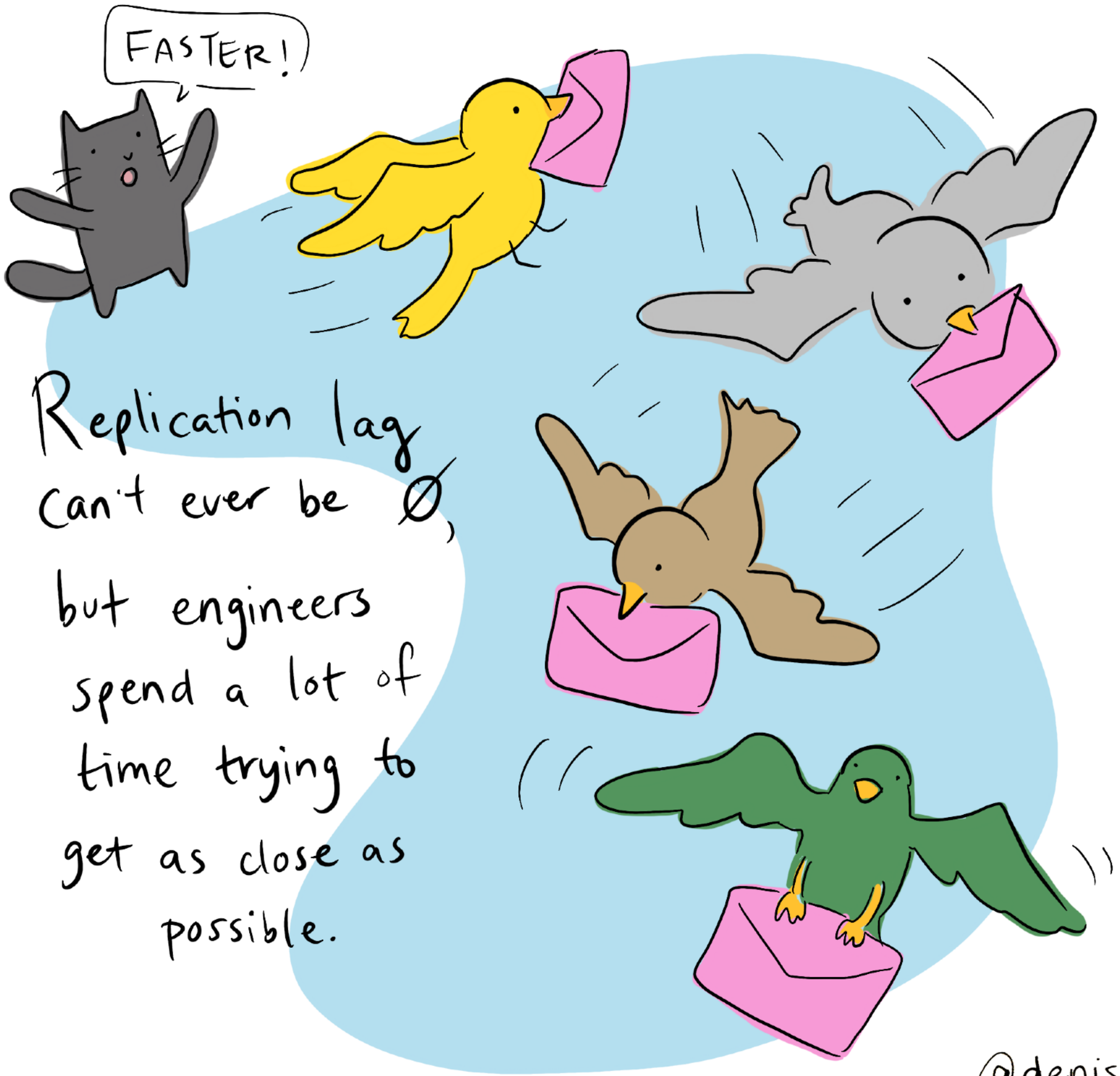
$t_0$  cat.state = 'hungry'

$t_1$  cat.state = 'full'

All nodes must have $t_1$ if anyone showed $t_1$

This is really hard! Instant & universal replication.

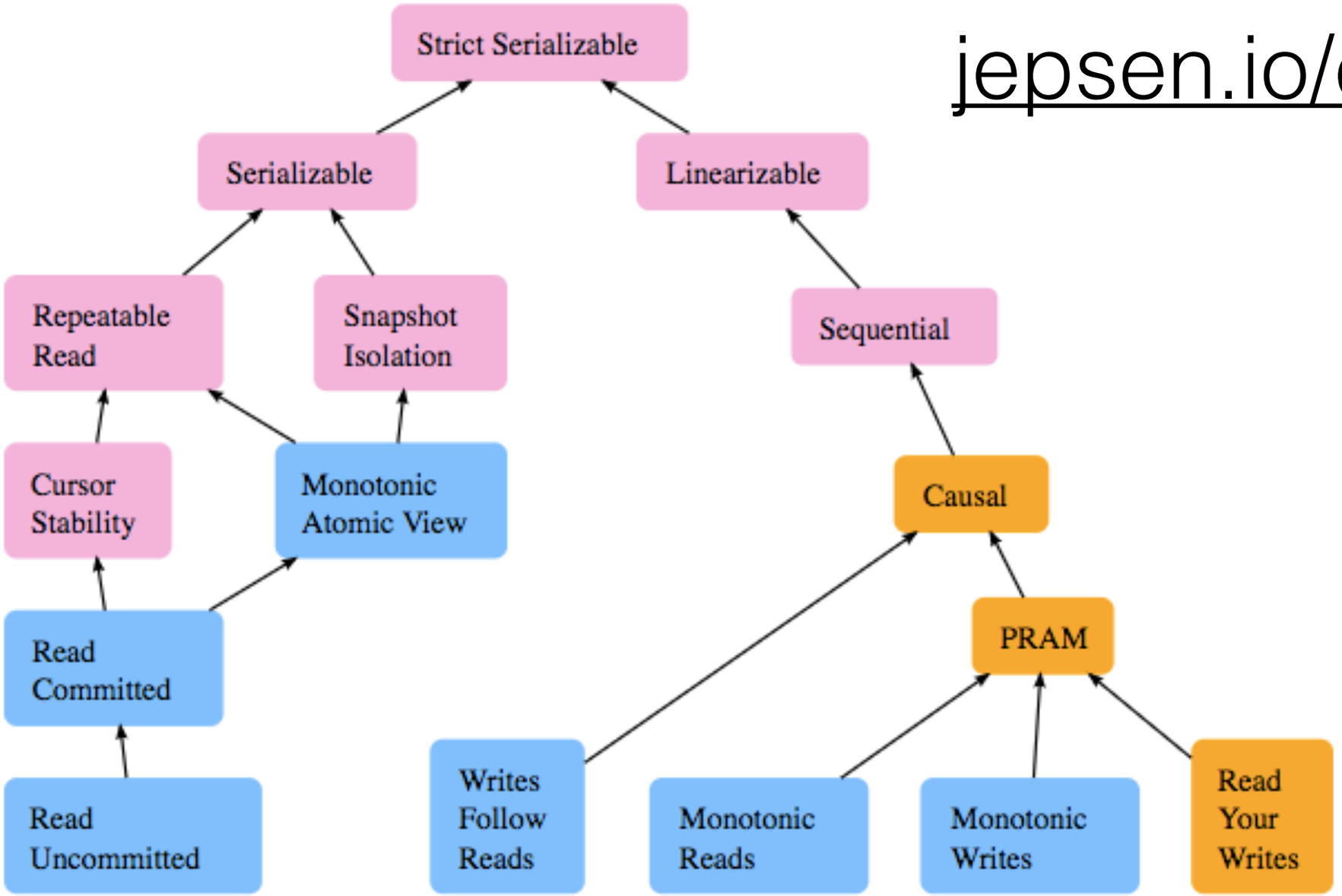
@deniseyu21


FASTER!
Replication lag
can't ever be Ø,
but engineers
spend a lot of
time trying to
get as close as
possible.

@deniseyu21

( BTW, eventual consistency doesn't count.

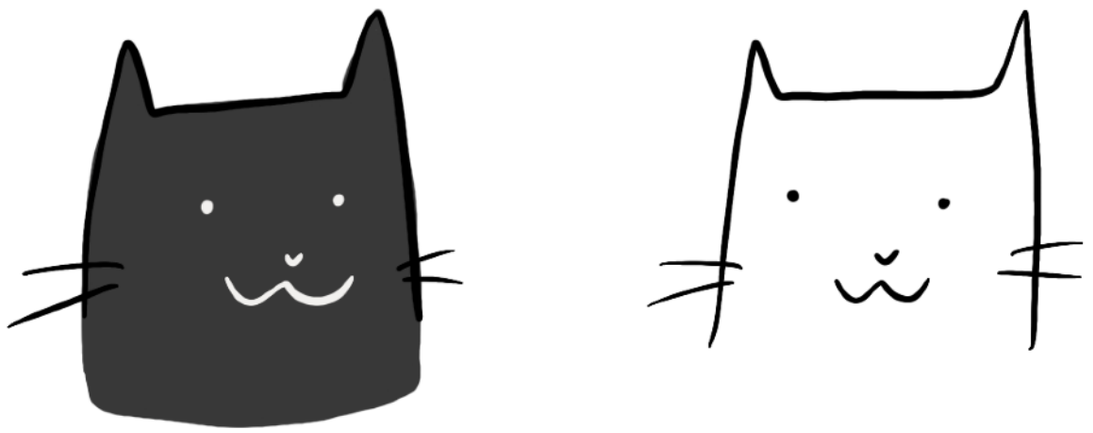
jepsen.io/consistency




Legend

| | |
|---|---|
| Unavailable | Not available during some types of network failures. Some or all nodes must pause operations in order to ensure safety. |
| Sticky Available | Available on every non-faulty node, so long as clients only talk to the same servers, instead of switching to new ones. |
| Total Available | Available on every non-faulty node, even when the network is completely down. |

# A IS FOR AVAILABILITY



We tend to think of availability as a binary state. BUT — reality is much messier!

Because LATENCY

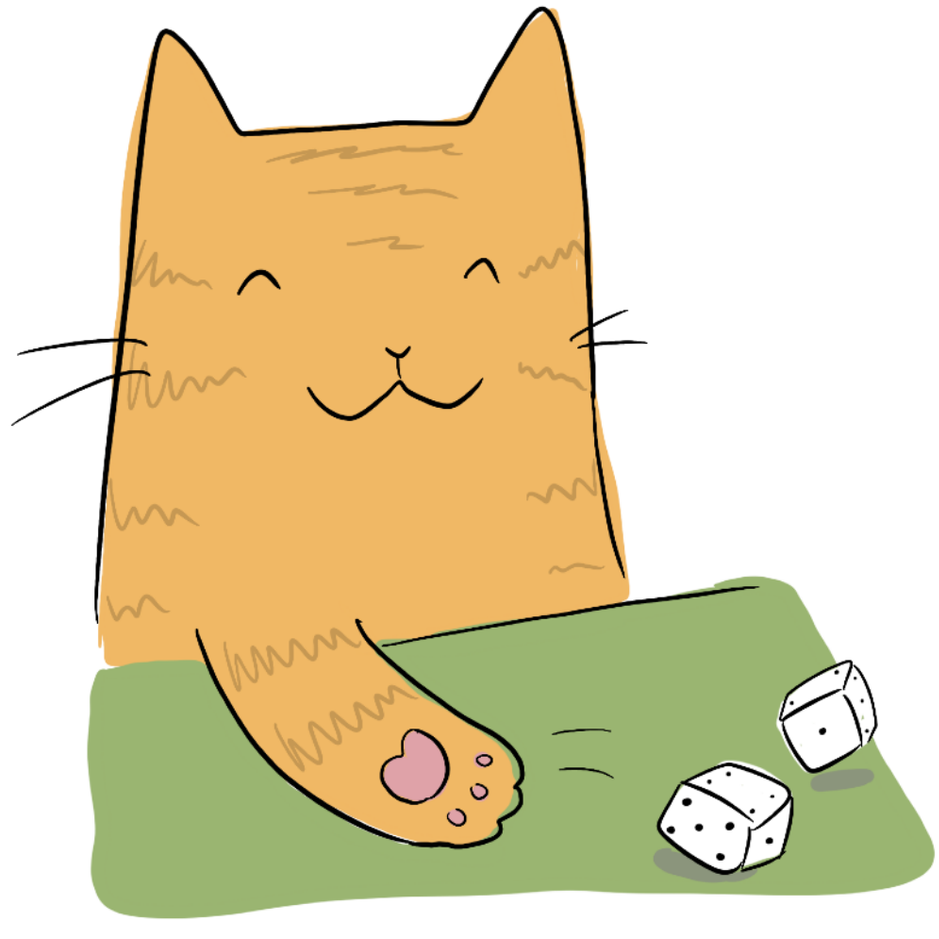
@deniseyu21

How can we know if a node is unresponsive...
or just slow?

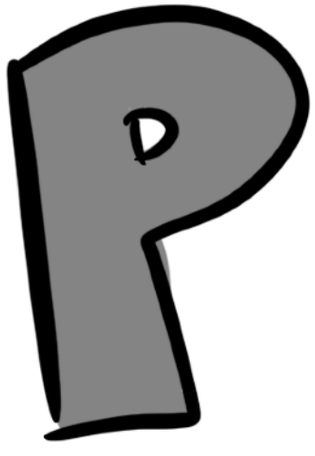

Network latency wasn't part of the original CAP formulation.

@deniseyu21

Determining a timeout limit is a very scientific process



yu 21



# IS FOR PARTITION TOLERANCE

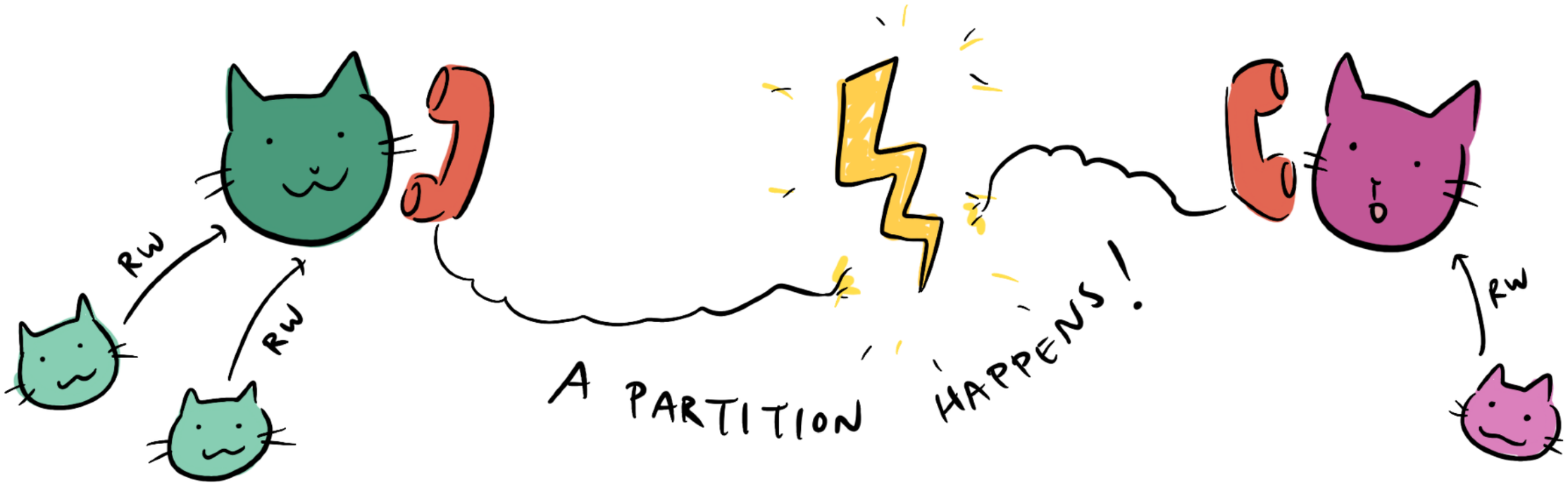
Network partitions occur when network connectivity between two datacenters (running your nodes!) is interrupted!

# P IS FOR PARTITION TOLERANCE

During a partition, your nodes might as well be on opposite sides of a wormhole: there is no way to know what's happening on the other side.


@deniseyu21

# PROOF OF CAP THEOREM




## OPTION 1

Let clients keep R/W in both sides of split

✗ LINEARIZABILITY

## OPTION 2

Stop writing in one side until partition ends

✗ AVAILABILITY

@deniseyu21

PARTITION TOLERANCE

Network partitions are inevitable.

How inevitable?

In the first year of a Google cluster's life, it will experience

5 rack failures

3 router failures

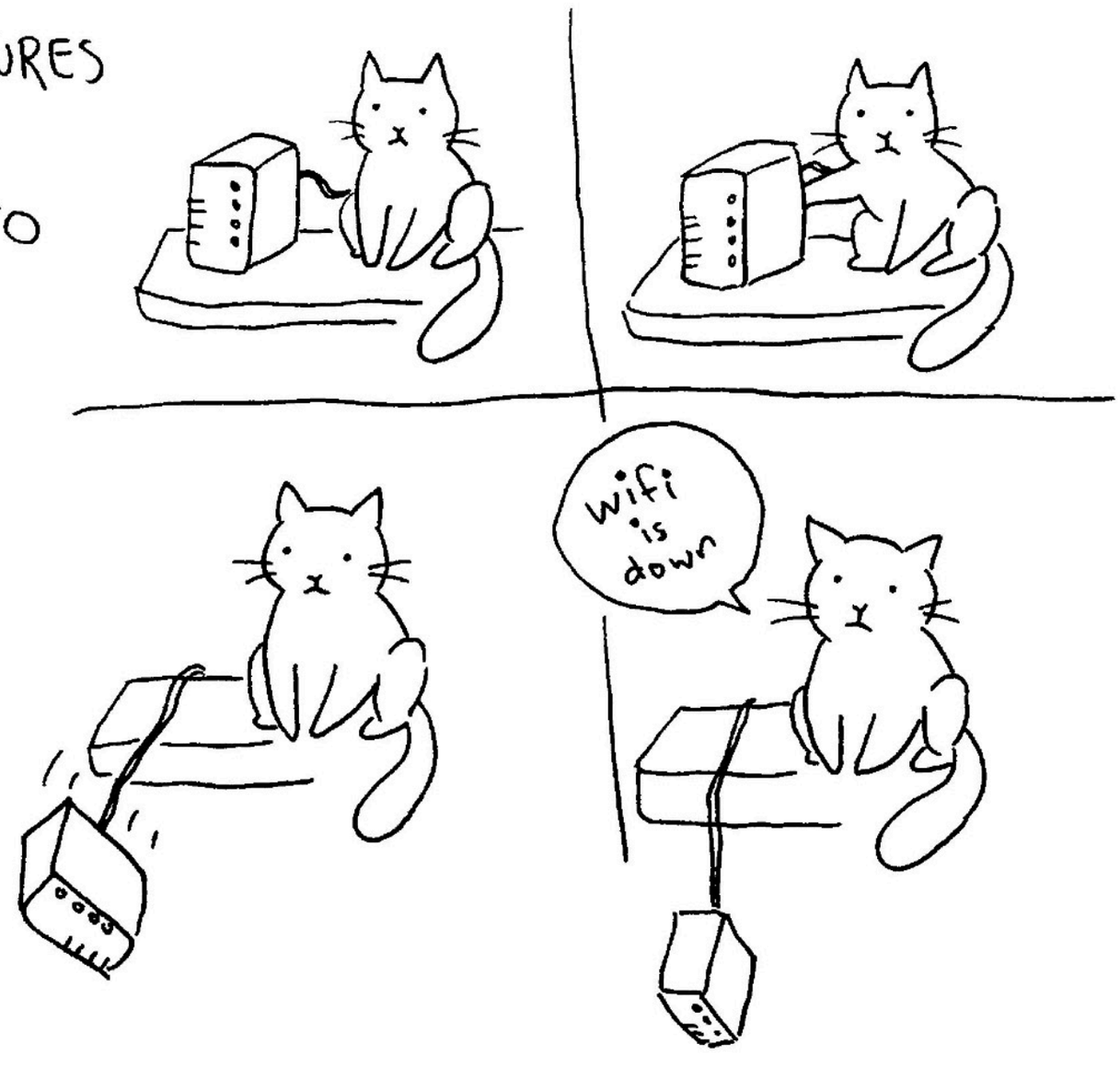
8 network maintenances

(Jeffrey Dean)


@deniseyu21

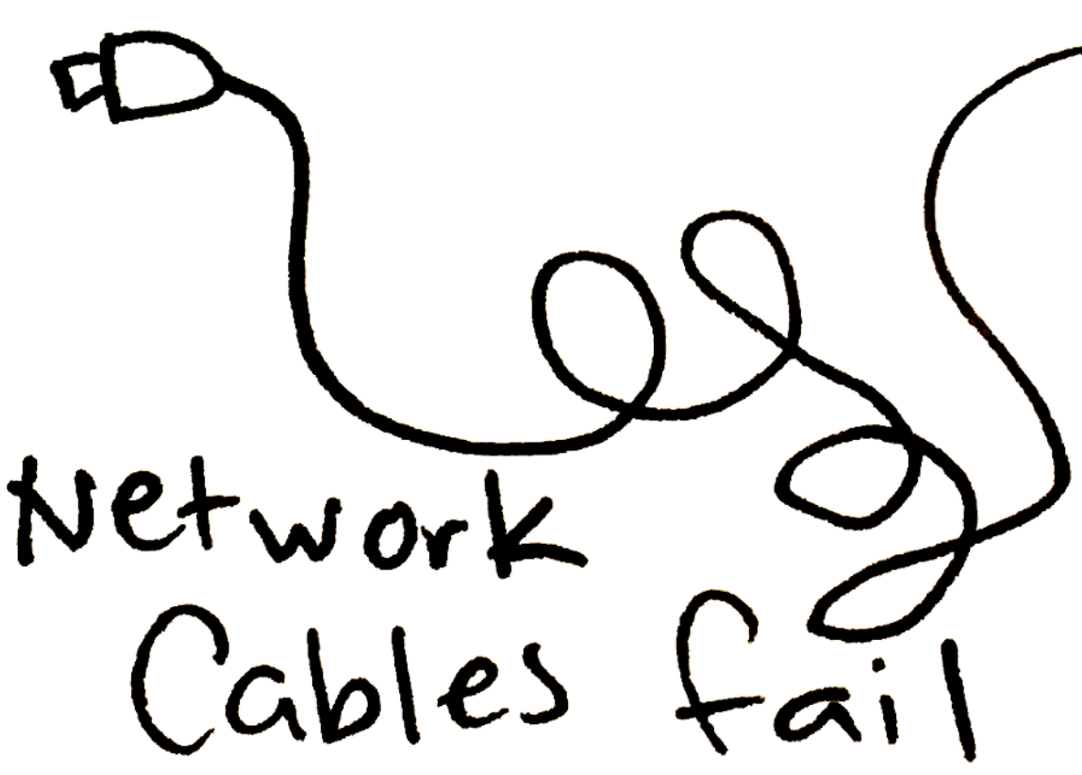
# Hardware will fail

ADVENTURES OF CATTO




@deniseyu21

Hardware will fail



Network Cables fail

@deniseyu21

Hardware will fail


@deniseyu21

POLICY —

# It's official: Sharks no longer a threat to subsea Internet cables

First known cable shark attacks were in 1985.

DAVID KRAVETS - 7/10/2015, 5:16 PM

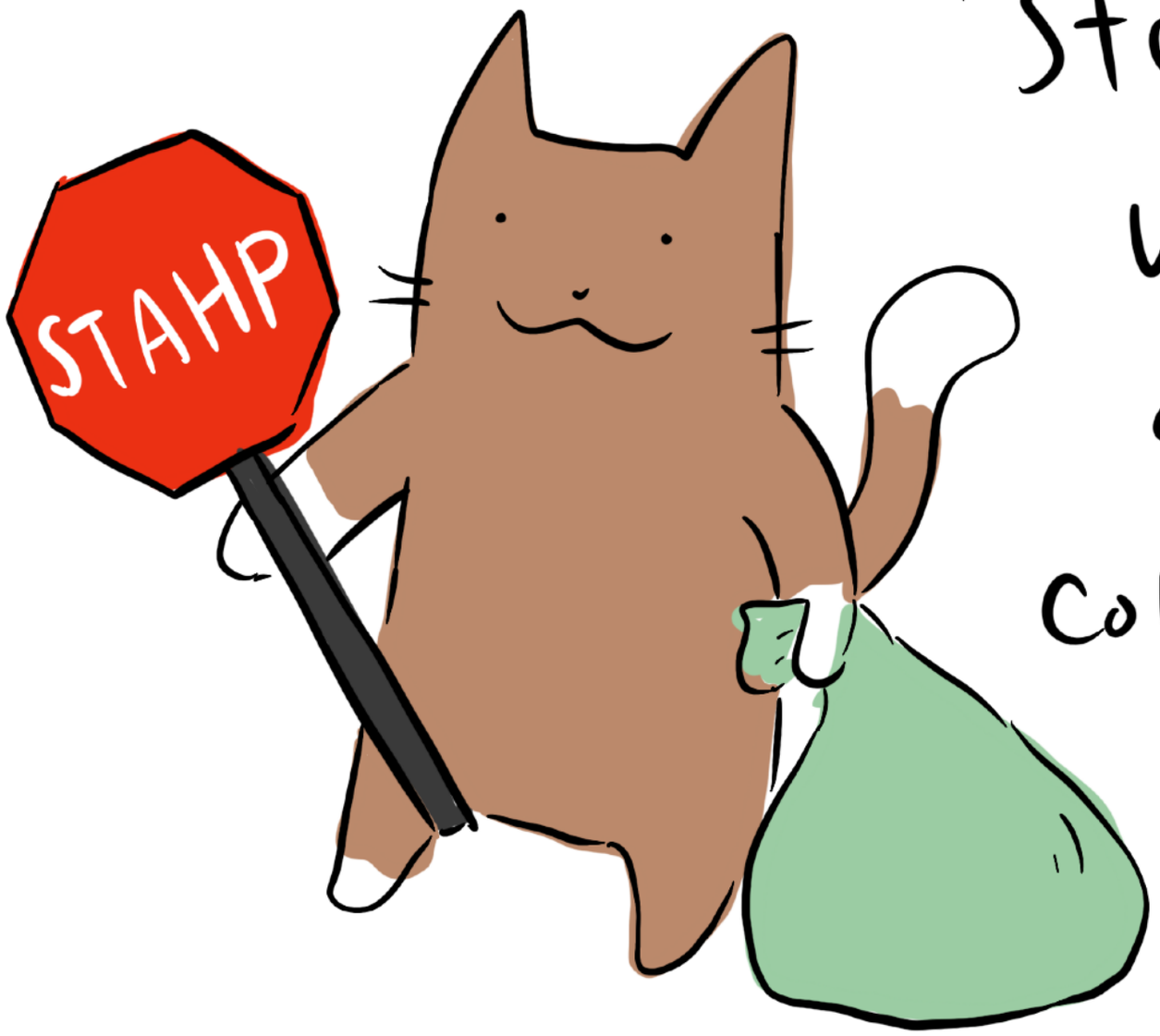
# Software will behave weirdly

"Bursty" VMs borrow resources from each other—

the Noisy Neighbor Problem


deniseyu21

Software will behave weirdly

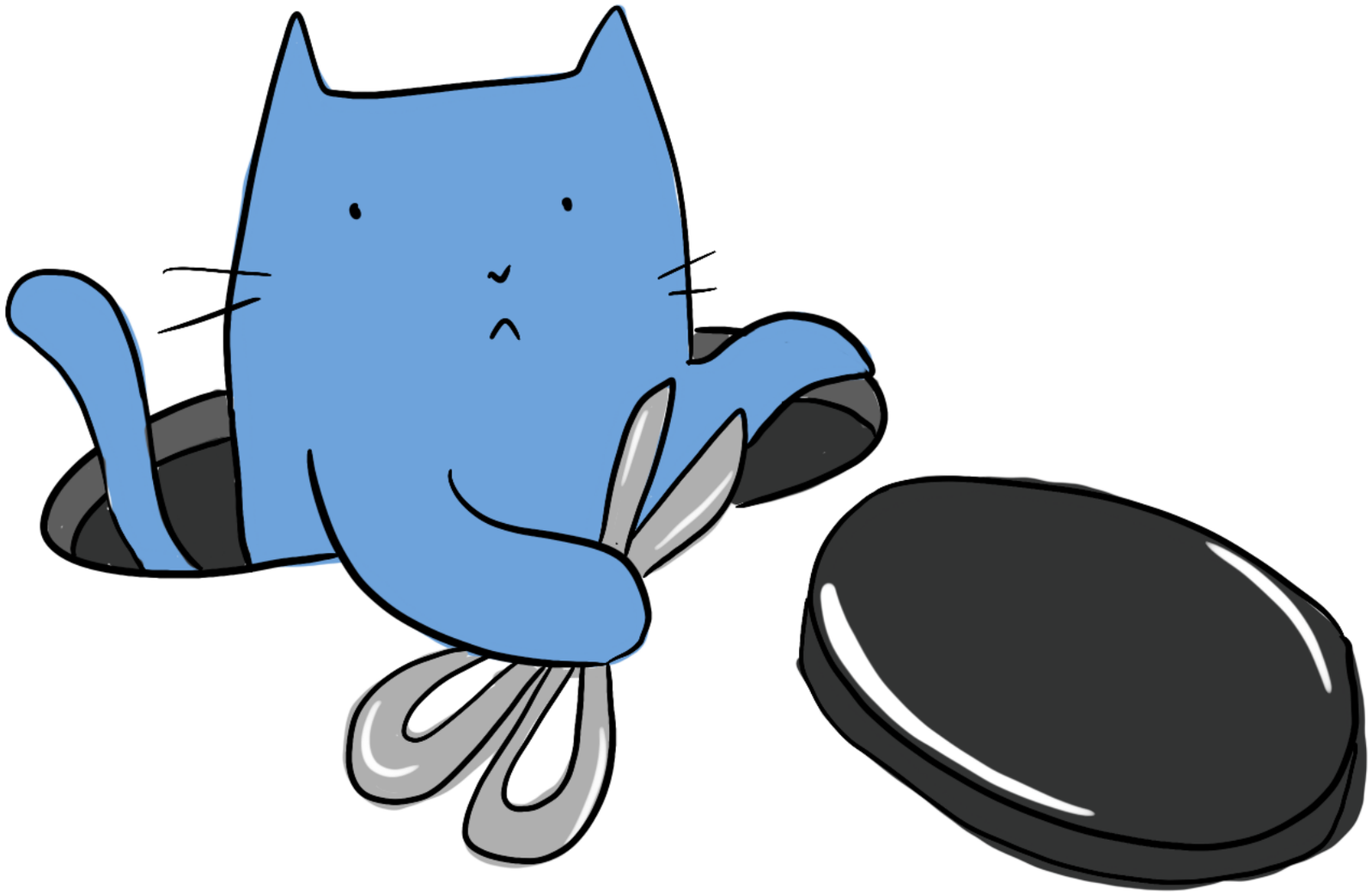



"Stop the WORLD" garbage Collection

@deniseyu21

# Network glitches

@deniseyu21



@deniseyu21


@deniseyu21

to manage uncertainty, we have mitigation strategies

# what is AUTOMATIC LEADER FOLLOWER ?

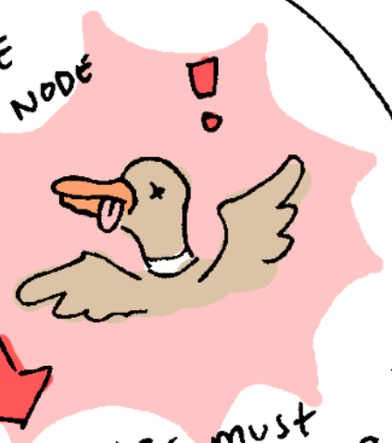
**LEADER-FOLLOWER*** is a common pattern for REPLICATING data across nodes in a cluster. If a follower node fails, it's NO BIG DEAL usually, because other nodes can continue to serve READ requests!


**STEP ONE**

Detect that a node failed! most databases have a default TIMEOUT (ex. 30s)

If a node has no activity for this time, it is considered to be offline

WHAT IF THE LEADER NODE FAILS?



the cluster must initiate a FAILOVER to choose a new leader!

**STEP TWO**

From the remaining followers, elect a new leader!

Becky's data is most up to date I vote Becky

Becky

YEH OK I'll do it

Some databases have a CONTROLLER NODE who chooses

He has ONE JOB!

**STEP THREE**

Tell the world about the new LEADER

I AM THE NEW LEADER!

yea Becky we know

THIS WAY

Traffic will be routed to the new leader for all future write requests

* You may have encountered "master-slave replication". I prefer not to use this term, because as technologists, our language matters. We should choose terms that are inclusive & don't cause harm.

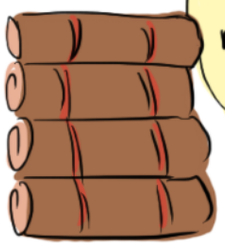
@deniseyu21
sketch-ops.tumblr.com

# How does RAFT work?

RAFT is a CONSENSUS ALGORITHM

a process for getting multiple machines to "agree"!

used in many projects in the real world - ex. etcd



- RAFT doesn't stand for anything. It's a bunch of logs tied together.

**WHY?** Distributed systems come with uncertainty. Achieving consensus is important so data can be replicated!

## WHAT ABOUT NETWORK PARTITIONS?!

In a partition, if the leader falls on the smaller side, a new leader is elected within the majority side. When the partition ends, messages received by the minority side are discarded, and those nodes converge their state to match the majority's.

## WHEN A NEW WRITE HAPPENS:

(Any node can write!)

Message is forwarded to leader node

Message is queued



Followers copy leader


Leader "commits" the new data

& followers increment their state counter

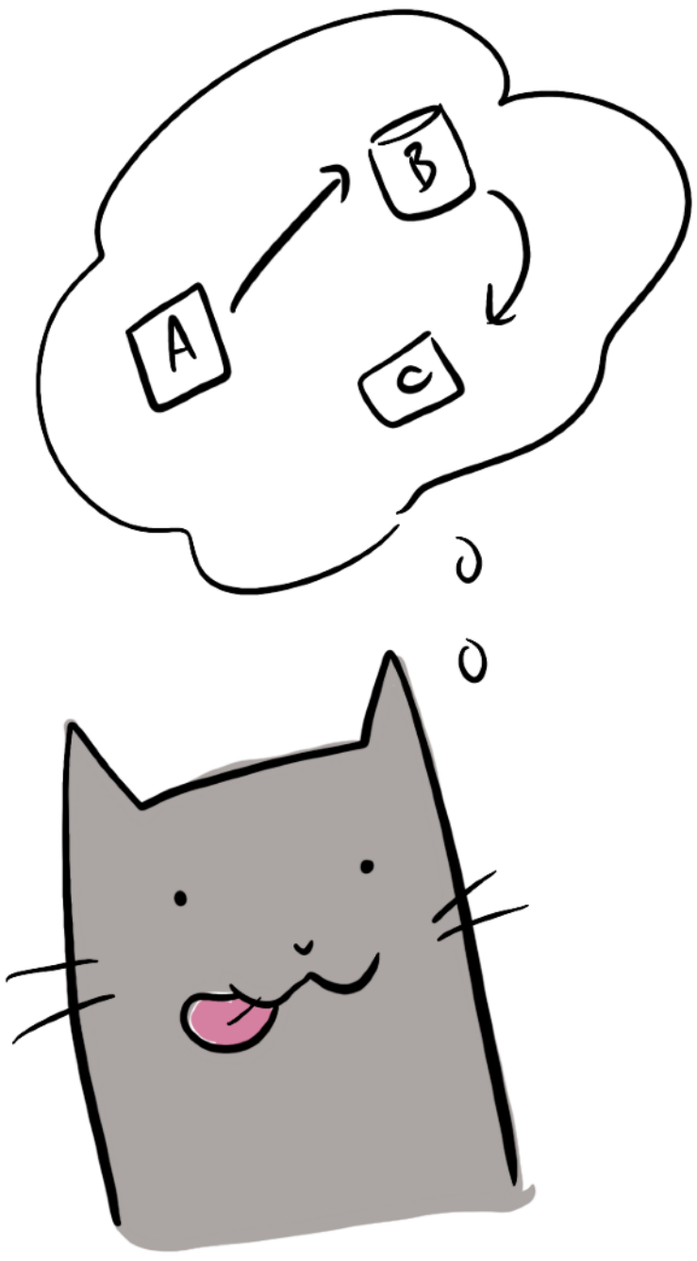
@deniseyu21 with @mt165

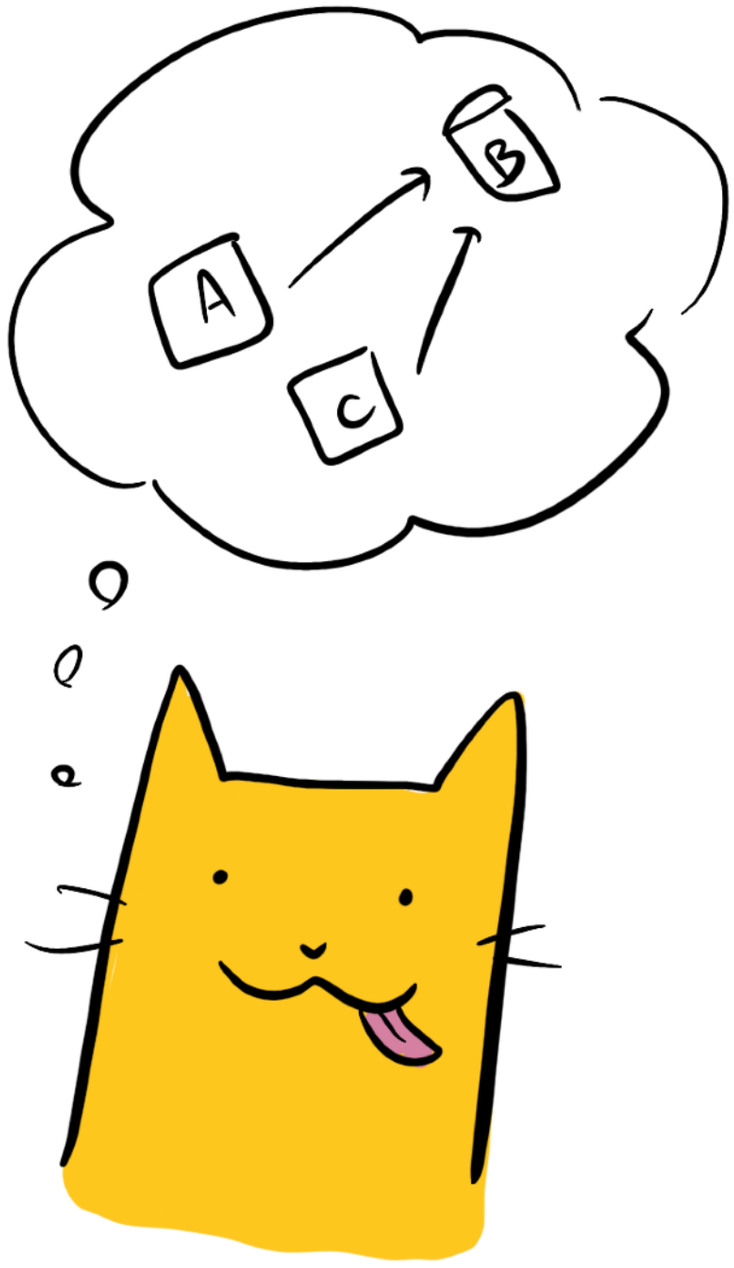
what is even harder than getting machines to agree?

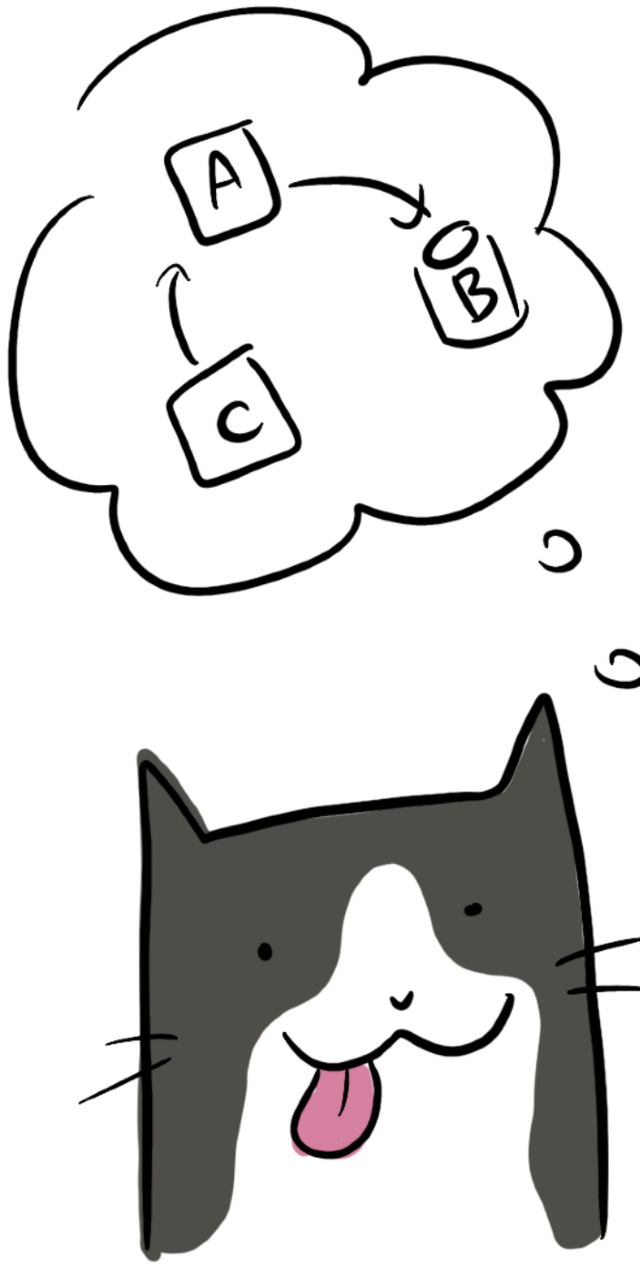
humans

Maybe we can learn from distsys in some social situations...


Mental models are hard to Compare,
which makes them hard to calibrate



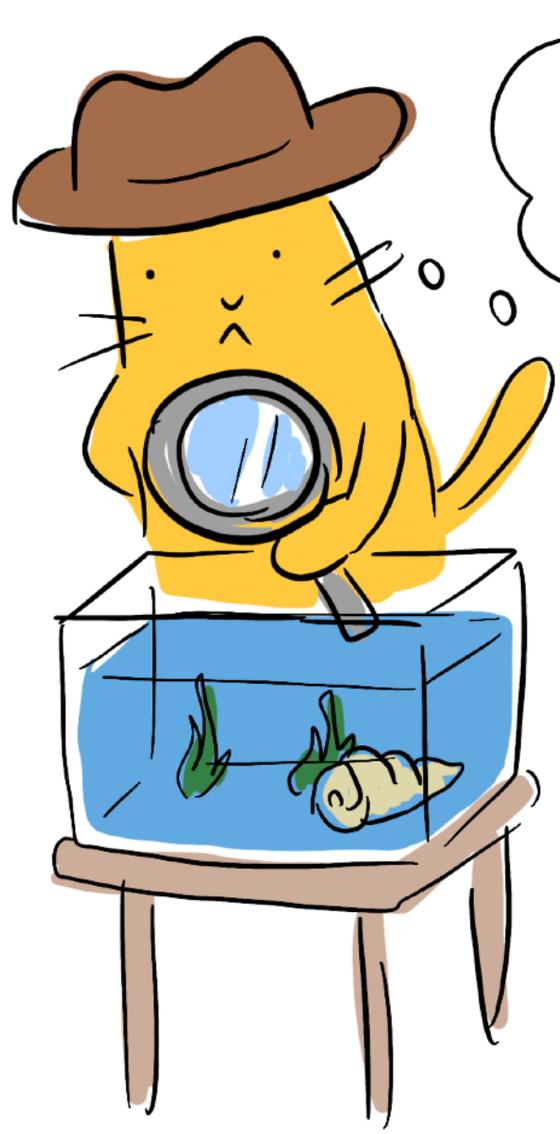




Mental models are hard to Compare, which makes them hard to calibrate

@deniseyu21

# INCIDENT ANALYSIS




This is the work of John Allspaw & his team

IS particularly great for mental model calibration

@deniseyu21


Blameless discussions optimized for learning

@deniseyu21

what happens when we get it wrong?


Nickolas Means,
"Who destroyed Three Mile Island?"

"All the News That's Fit to Print"

# The New York Times

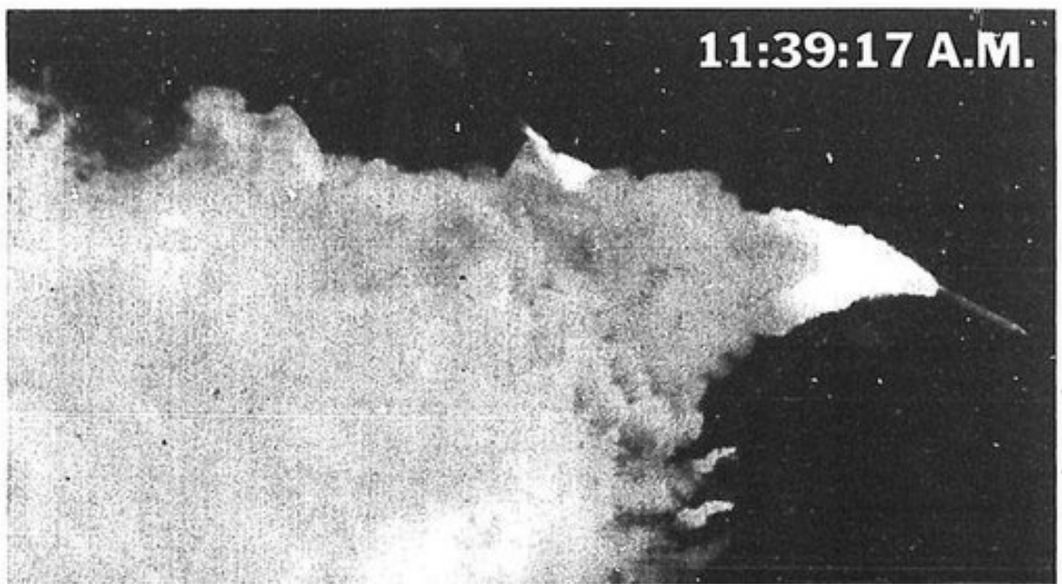
**Late Edition**
Weather: Partly cloudy and cold today, chance of snow; chance of snow tonight. Partly cloudy, cold tomorrow. Temperatures: today 27-30, tonight 13-19; yesterday 14-23. Details, page C19.

VOL.CXXXV . . . No. 46,669  NEW YORK, WEDNESDAY, JANUARY 29, 1986

60 cents beyond 75 miles from New York City, except on Long Island. 30 CENTS

# THE SHUTTLE EXPLODES

## *6 IN CREW AND HIGH-SCHOOL TEACHER ARE KILLED 74 SECONDS AFTER LIFTOFF*


### Thousands Watch A Rain of Debris

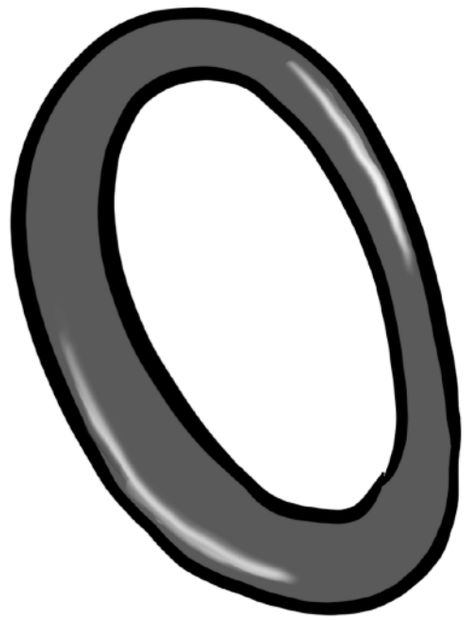
By WILLIAM J. BROAD
Special to The New York Times

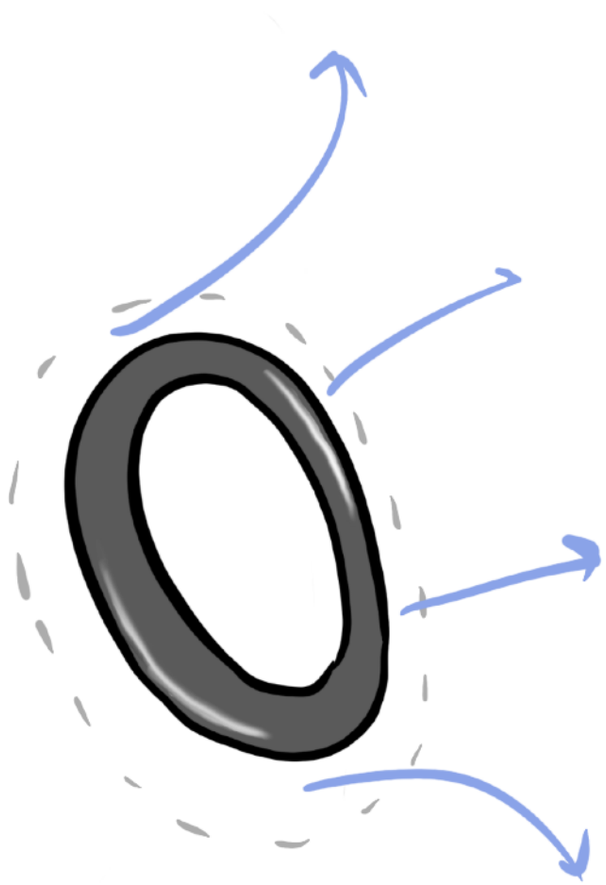
CAPE CANAVERAL, Fla., Jan. 28 — The space shuttle Challenger exploded in a ball of fire shortly after it left the launching pad today, and all seven astronauts on board were lost.

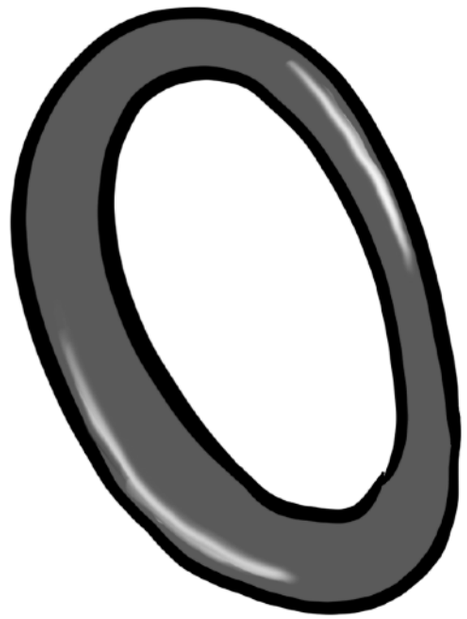
The worst accident in the history of the American space program, it was witnessed by thousands of spectators who watched in wonder, then horror, as the ship blew apart high in the air.

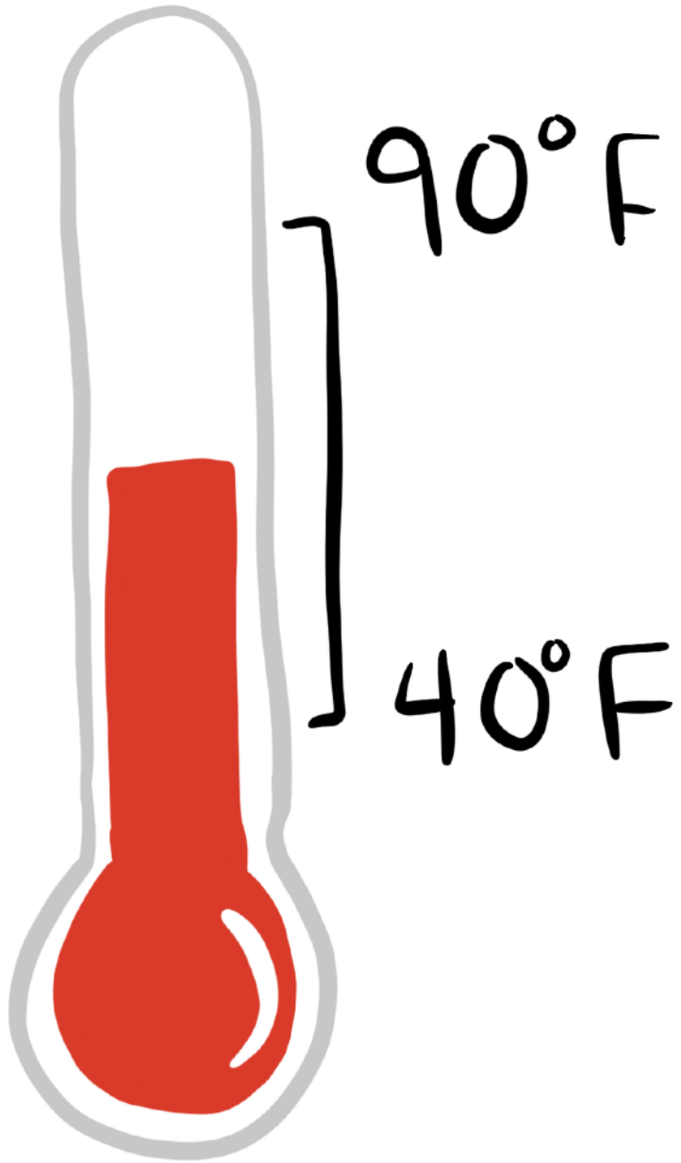
Flaming debris rained down on the Atlantic Ocean for an hour after the explosion, which occurred just after 11:39 A.M. It kept rescue teams from reaching the area where the craft would have fallen into the sea, about 18 miles offshore.

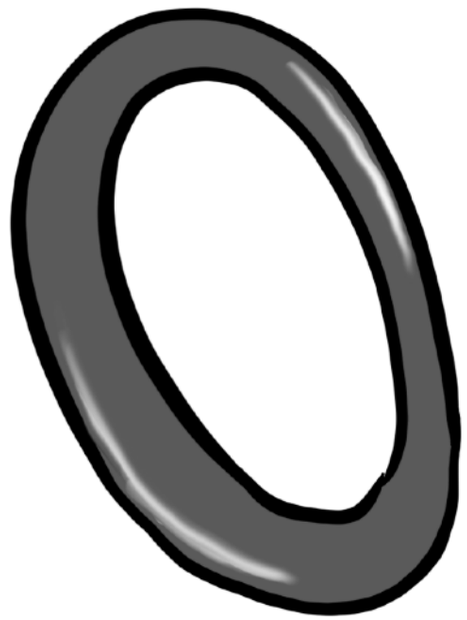
It seemed impossible that anyone could have



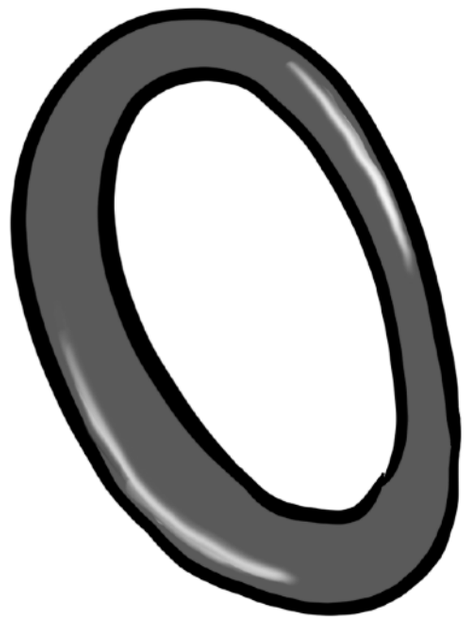


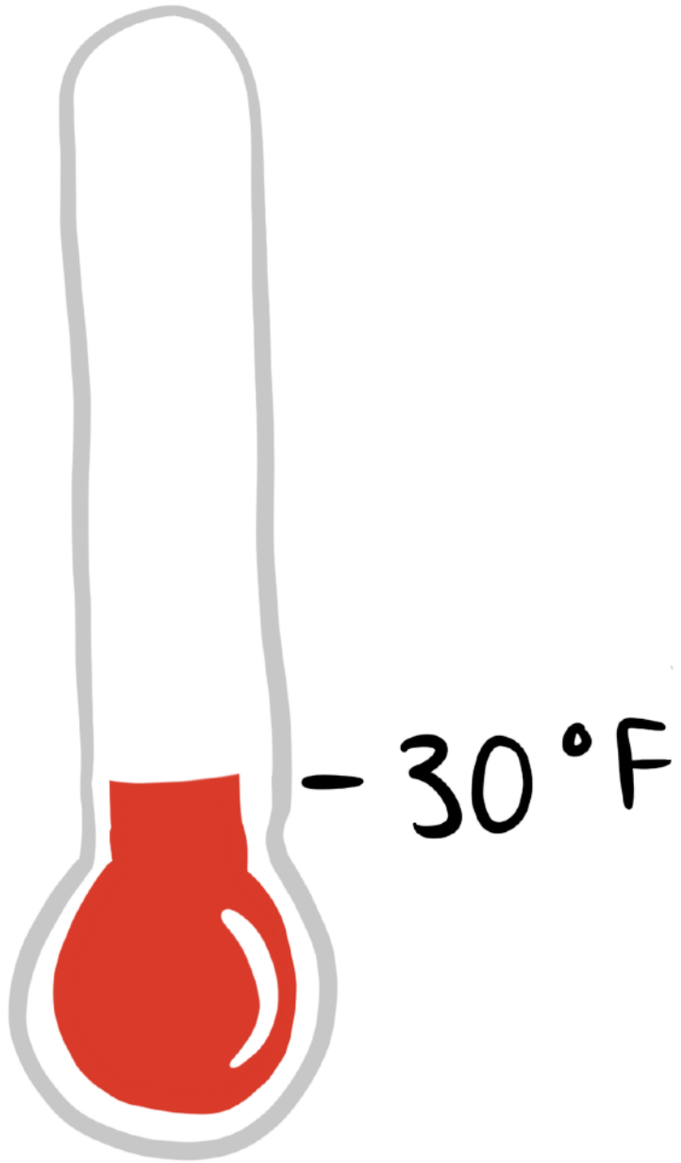


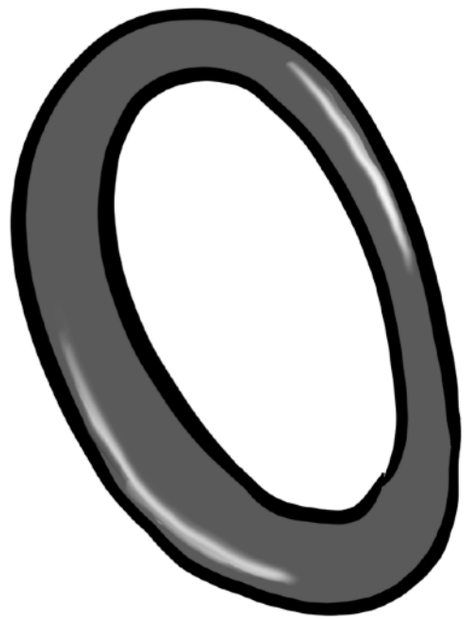

90°F
40°F




-35°F




-30°F

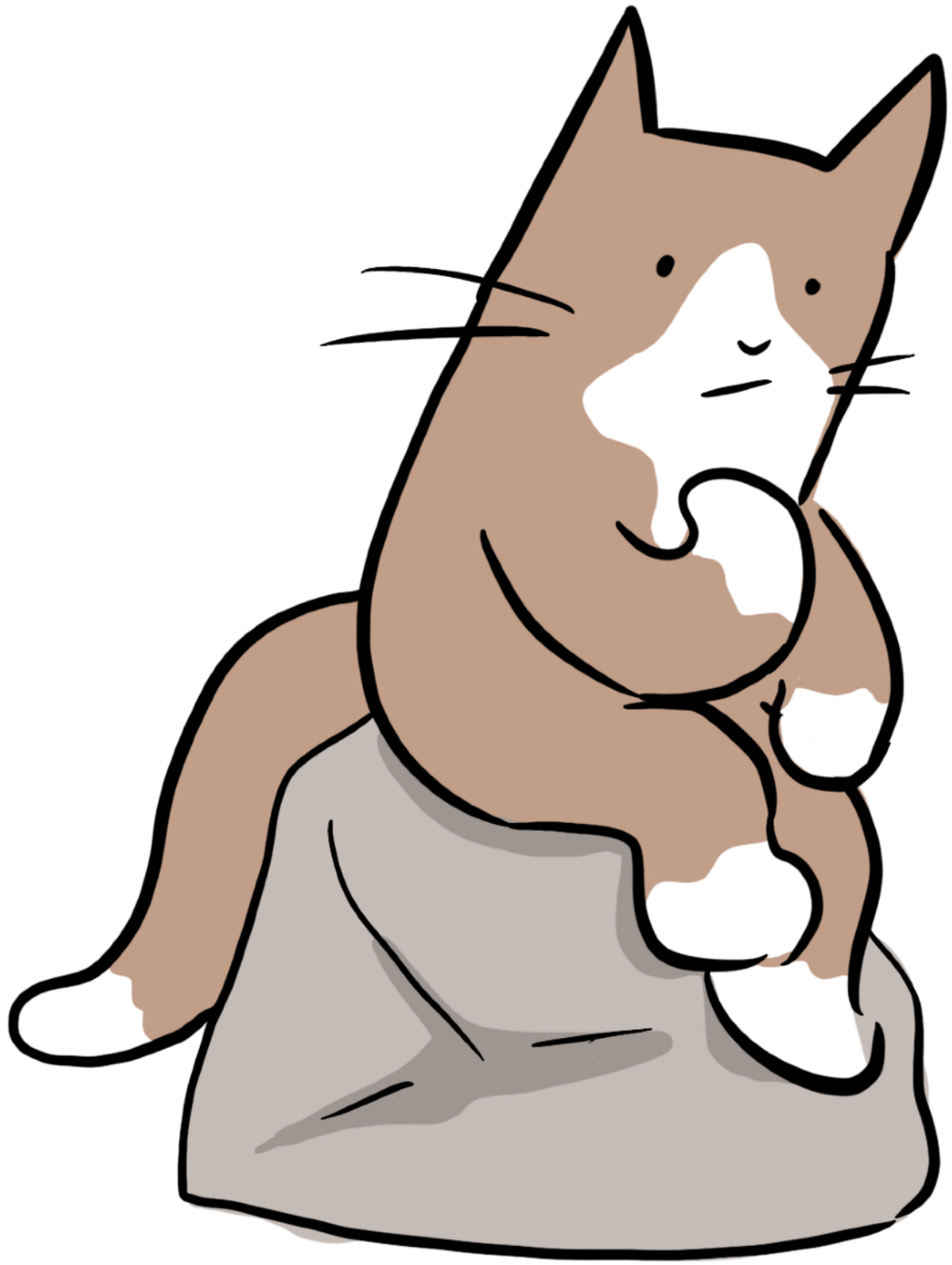


- 28°F

# the normalization of deviance

@foone

foone.wordpress.com/2019/02/14/normalization-of-deviance



Don't accept HUMAN ERROR as the root cause. Dig deeper!

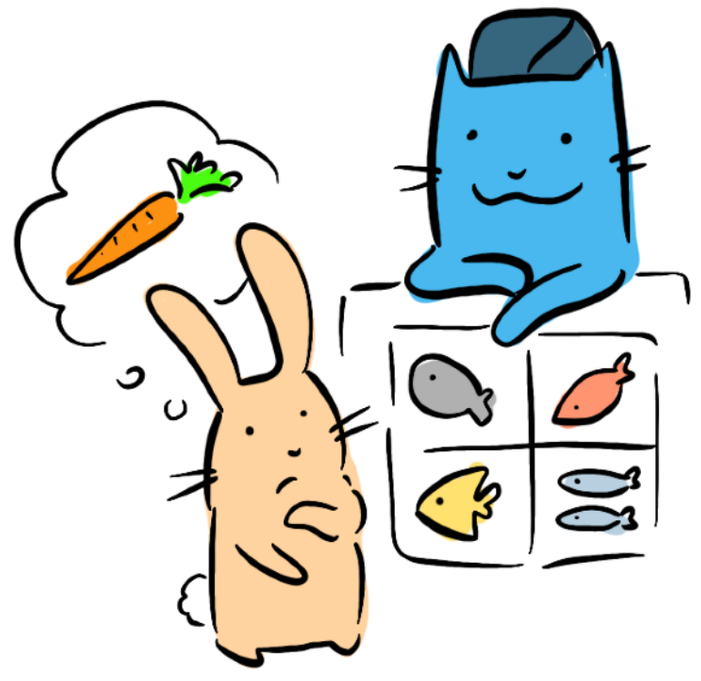
@deniseyu21

Unintuitive design?

☐ check if you do not not not not not not not wish to receive emails


Alert fatigue?

Not understanding the users' assumptions and needs?



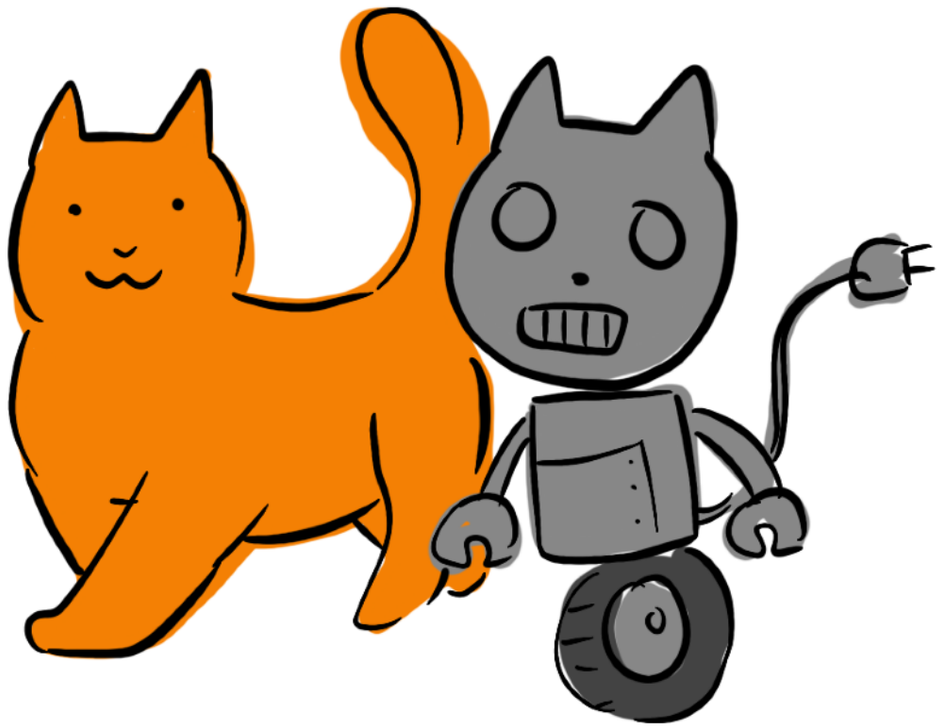
@deniseyu21

We have a better superpower:

EMPATHY

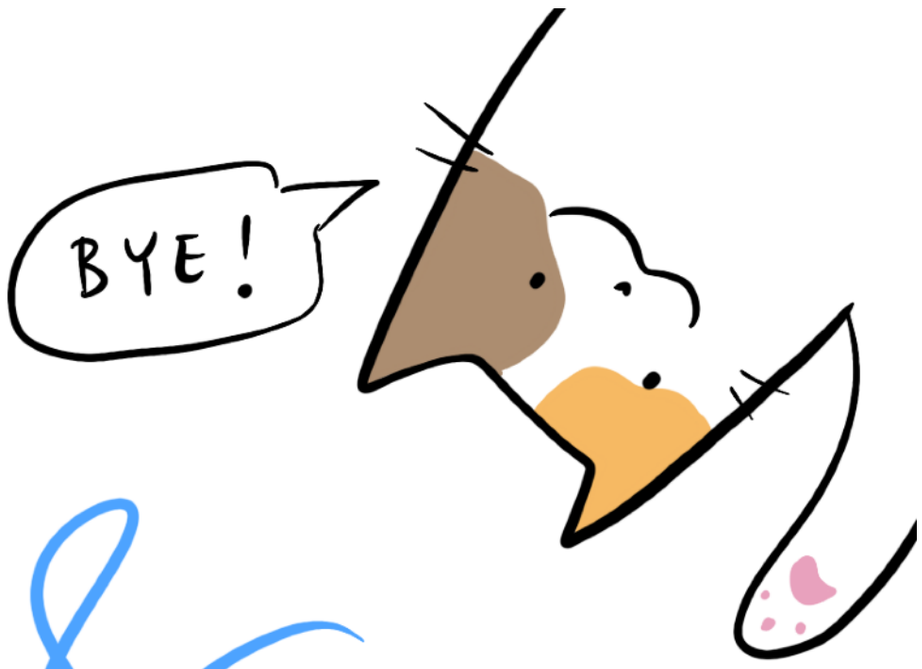
@deniseyu21

we owe it to our end users &
our teams to understand
& design for the whole system



including the fleshy human parts.

@deniseyu21




Slides &
References

denise.yu.io/srecon