why are DISTRIBUTED SYSTEMS so hard?

@deniseyu21

deniseyu.io

A long time ago,

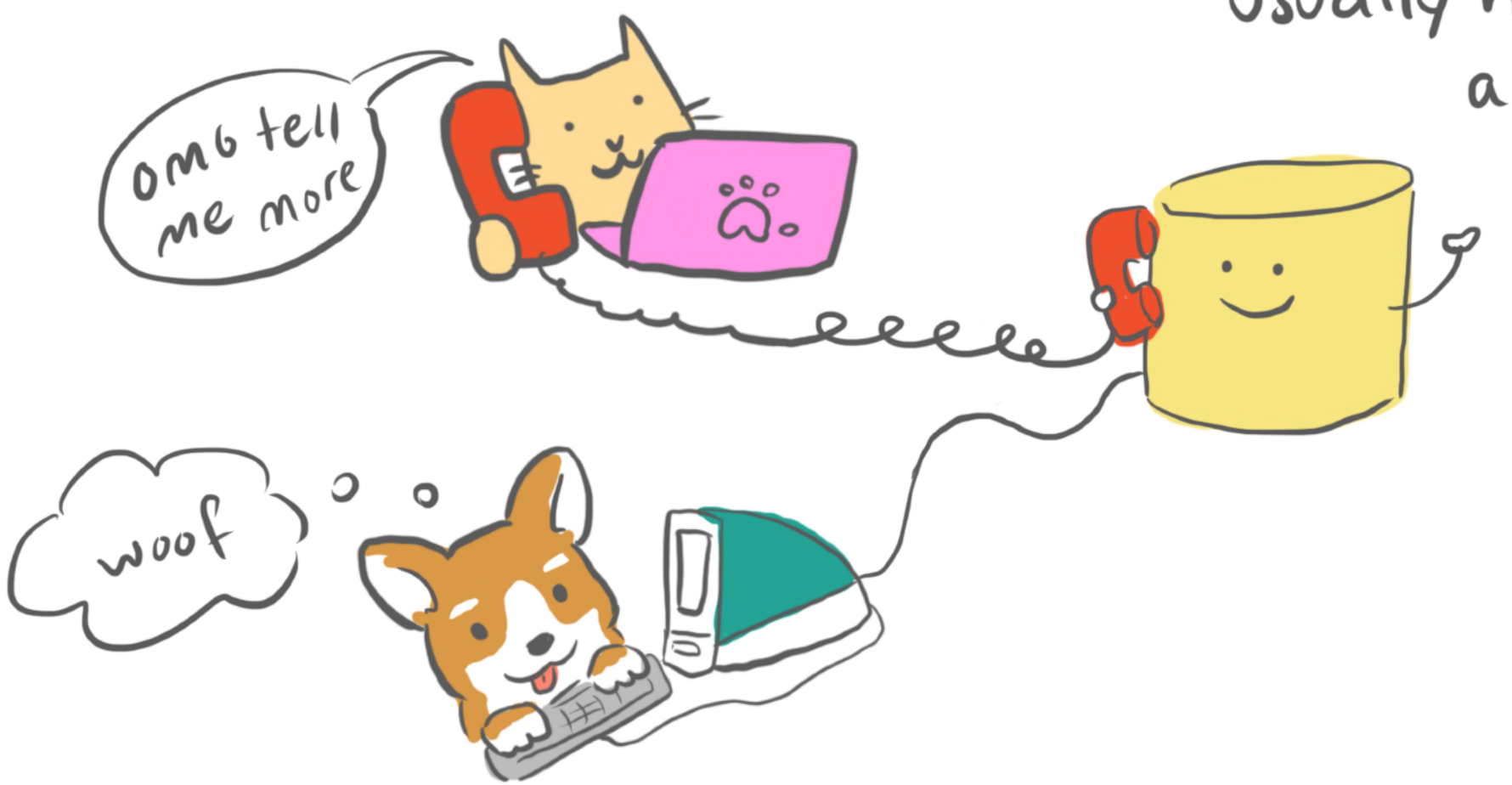
in a datacenter not too far away...

A long time ago,

in a datacenter not too far away...

All business applications talked to one database,

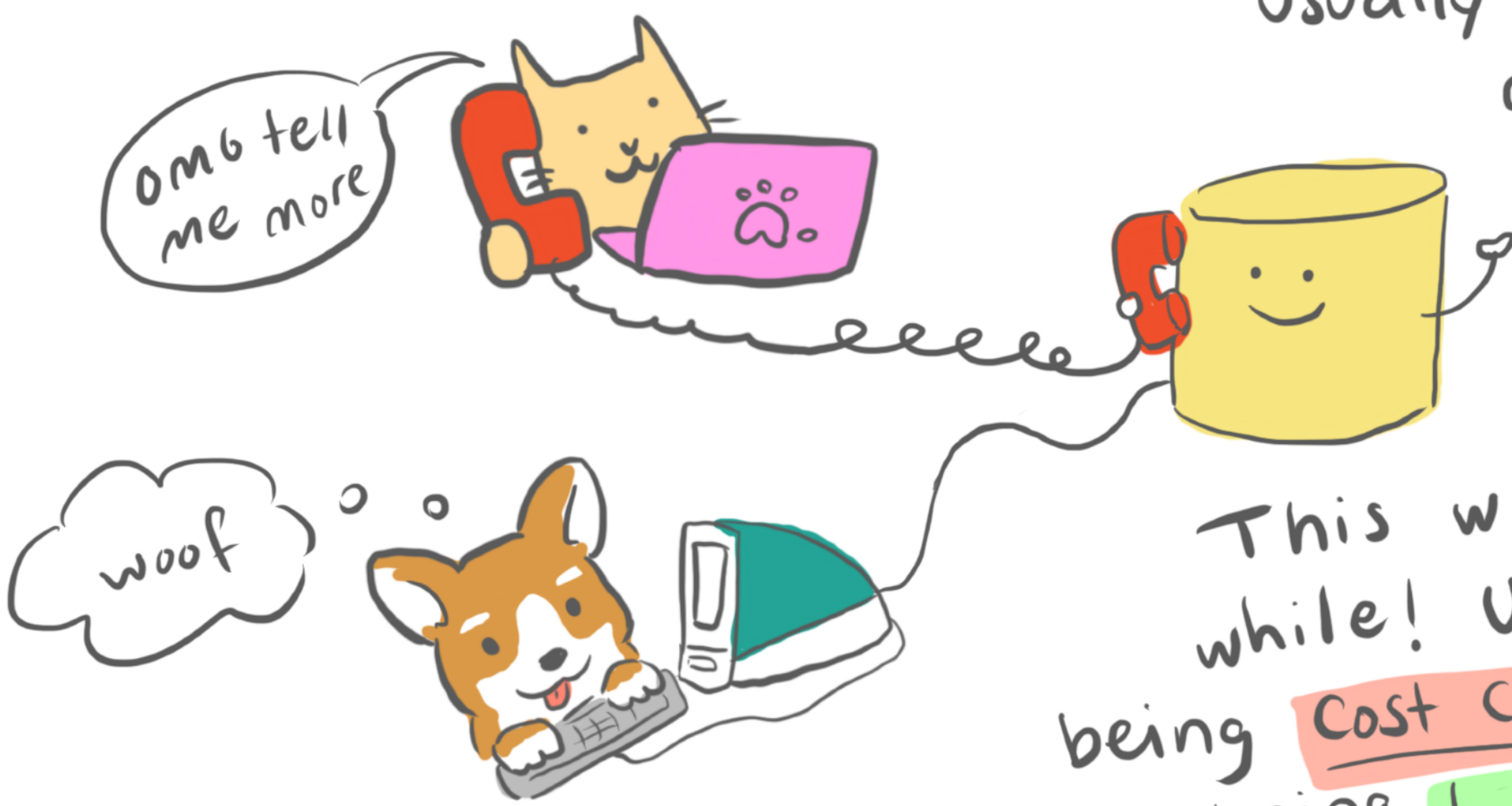
usually hosted on a company's own hardware.



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.

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

Business analysis
& data warehouses

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

Business analysis & data warehouses

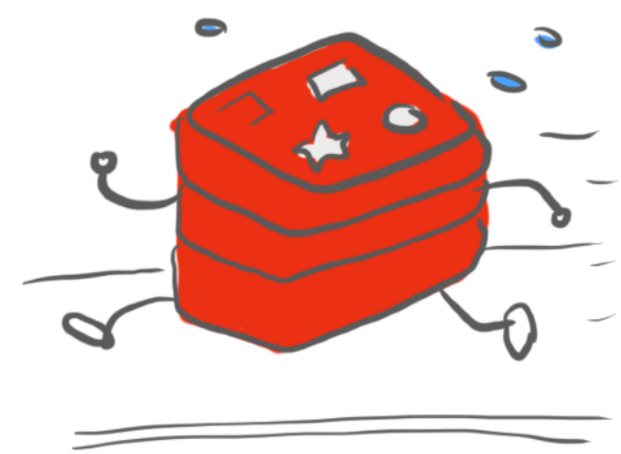
ML & Natural language processing

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...

So we scaled vertically...

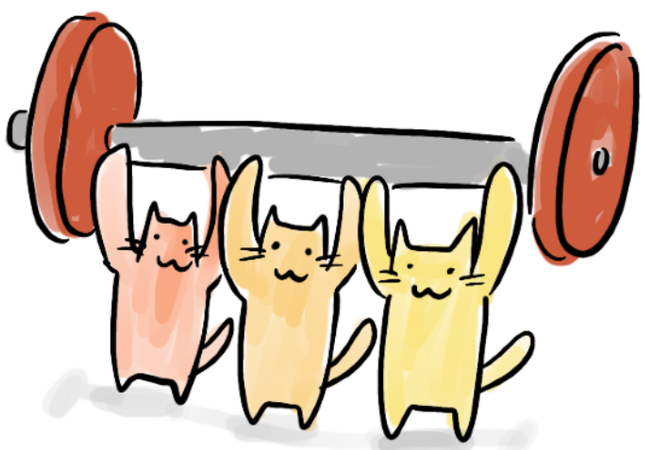
until unit economics (or physics) caught up.

@deniseyu21

@deniseyu21

REASONS TO HORIZONTALLY DISTRIBUTE:

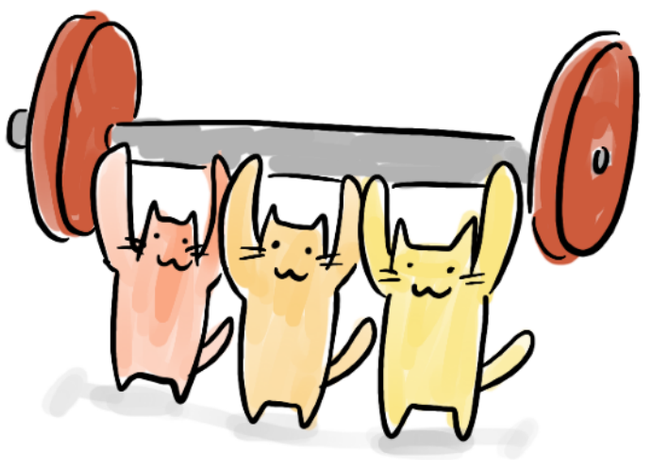
REASONS TO HORIZONTALLY DISTRIBUTE:



Scalability:

one machine cannot handle request or data size

REASONS TO HORIZONTALLY DISTRIBUTE:



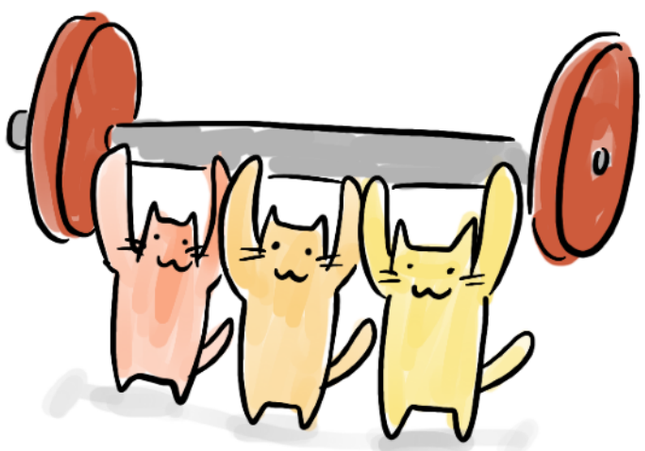
Scalability:

one machine cannot handle request or data size

Availability:

if one machine goes down, others keep working

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

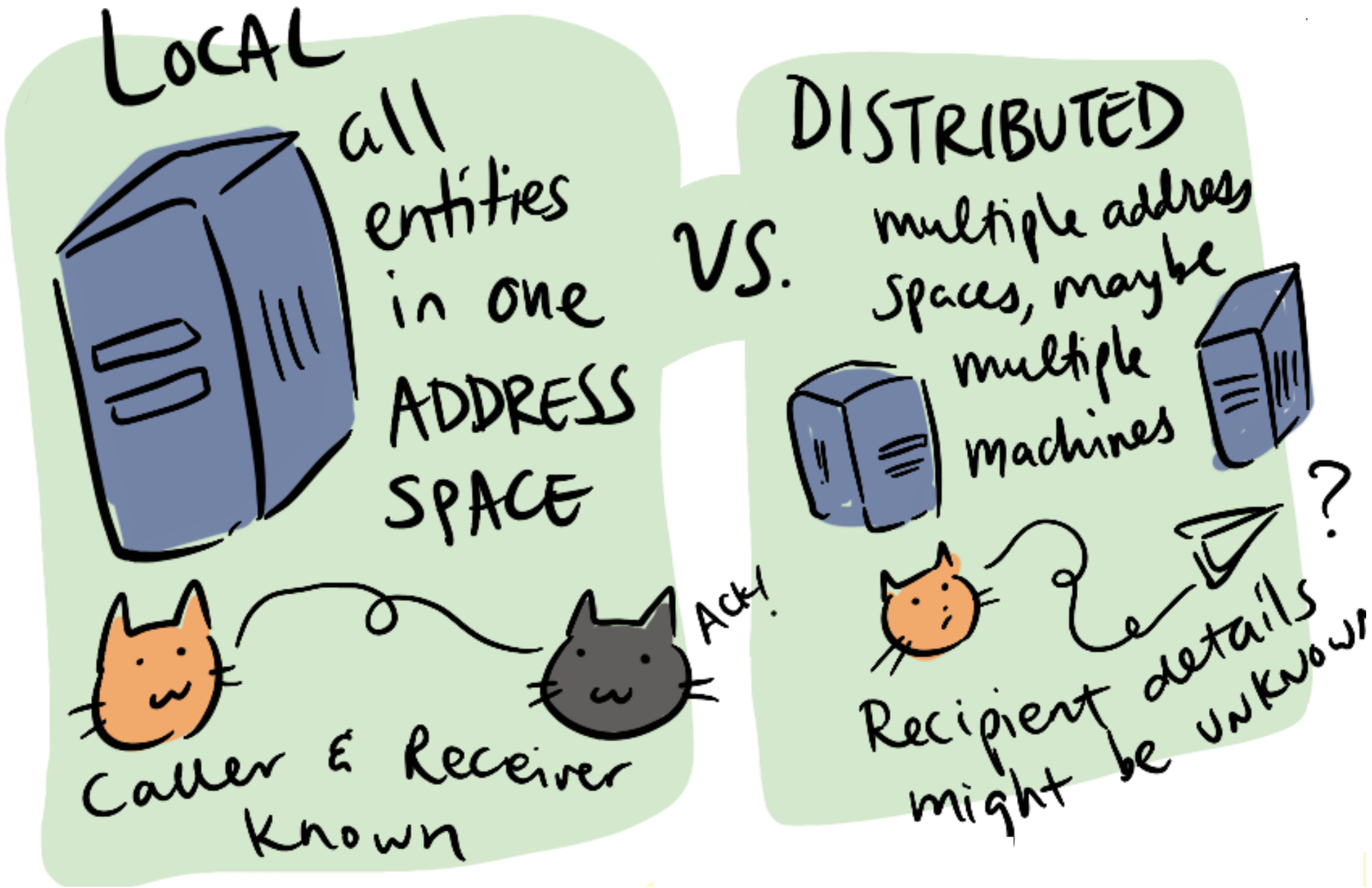
You may have heard the term "shared nothing" architecture:

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

This is mostly sensible, but it does force design trade-offs.

@deniseyu21

what does it actually mean to run a distributed system?


LOCAL
all entities in one ADDRESS SPACE
Ack!
Caller & Receiver Known
VS.
DISTRIBUTED
multiple address spaces, maybe multiple machines
?
Recipient details might be unknown

@deniseyu21

a note on distributed computing

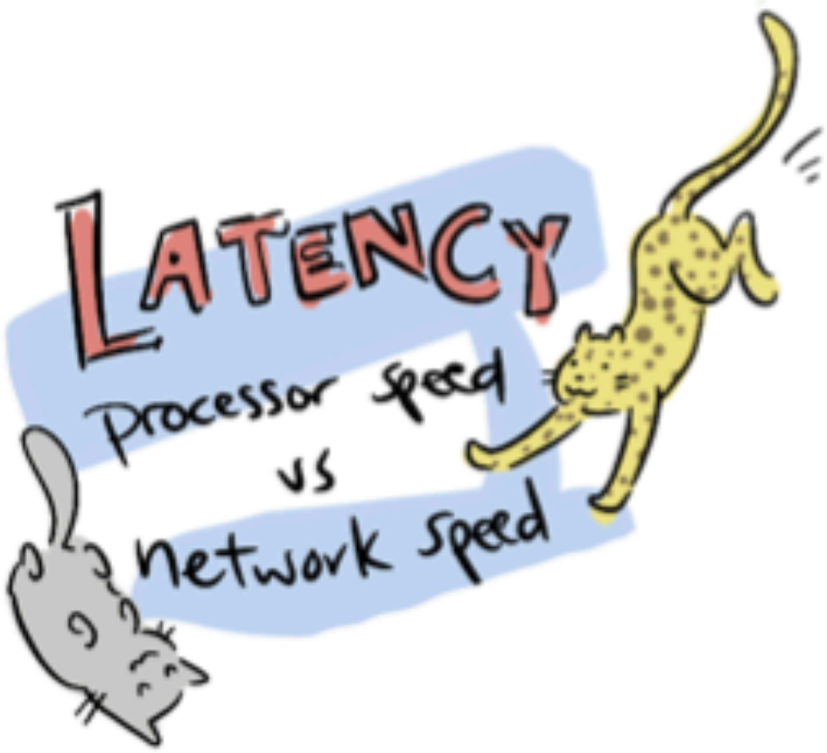
1994

@deniseyu21

THE HARD PARTS!

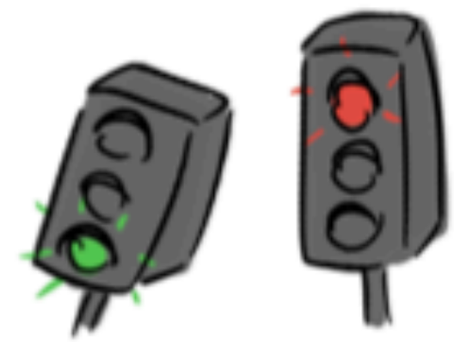
LATENCY

processor speed
vs
network speed



MEMORY ACCESS

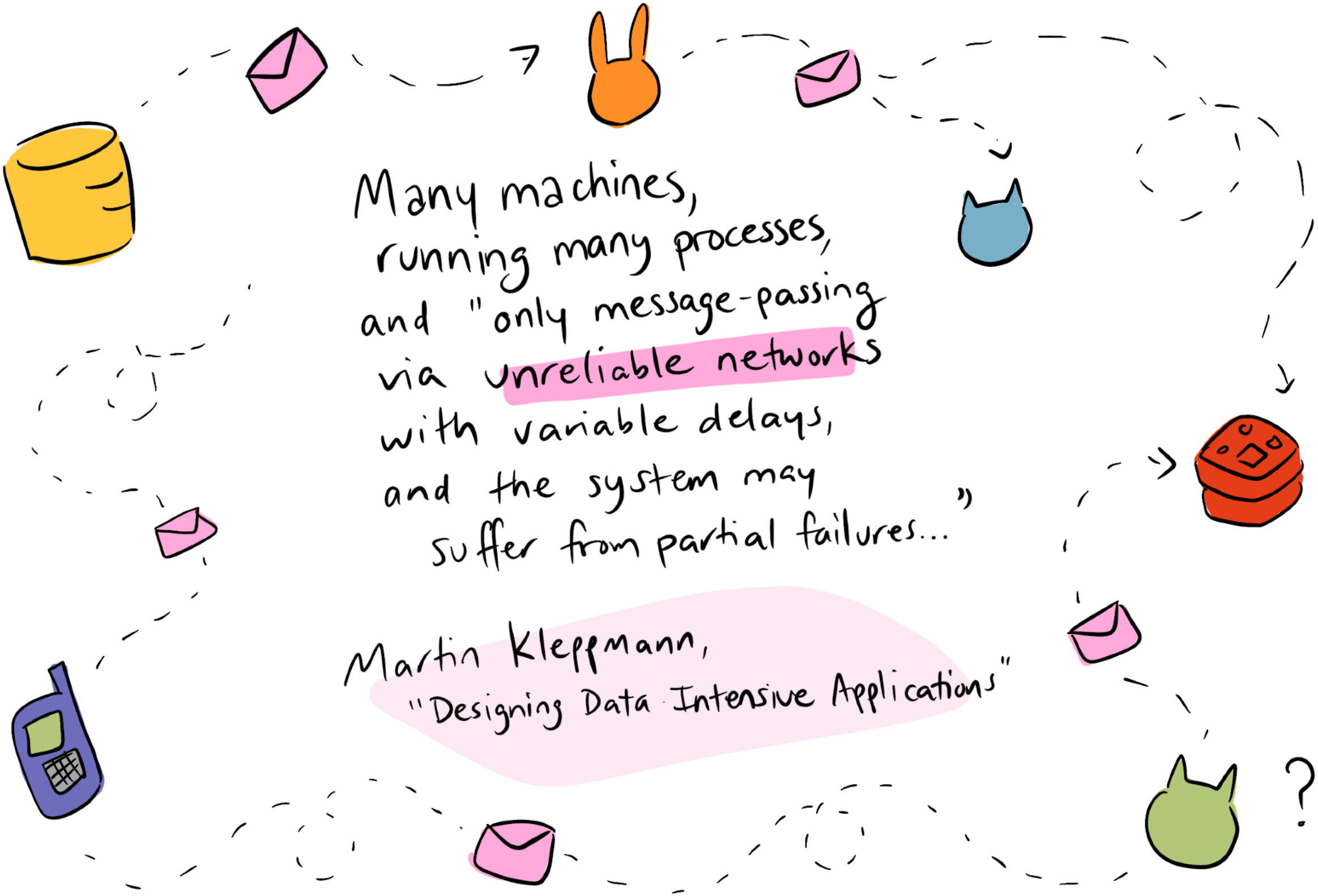
Pointers are only valid in their own memory space



PARTIAL FAILURES

↳ inevitable!

@deniseyu21


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

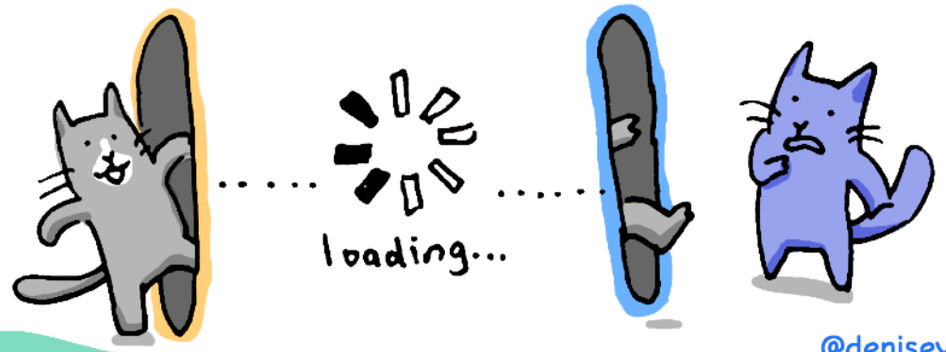
the 8 Fallacies of Distributed Computing

@deniseyu21

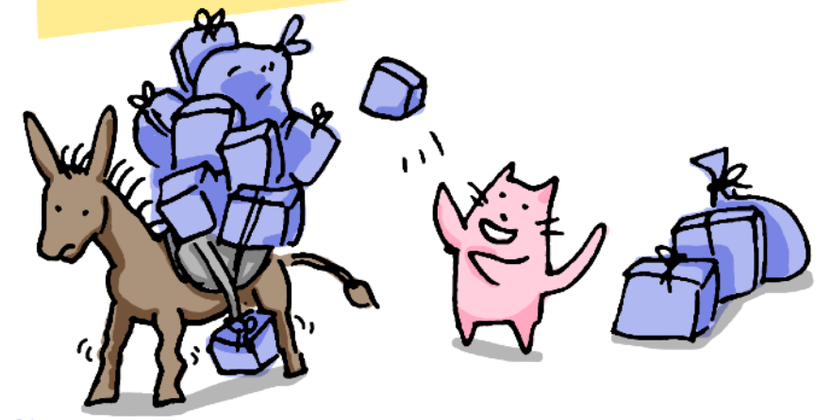
Originally formulated by L. Peter Deutsch & Colleagues at Sun Microsystems in 1994; #8 added in 1997 by James Gosling

① The network is reliable

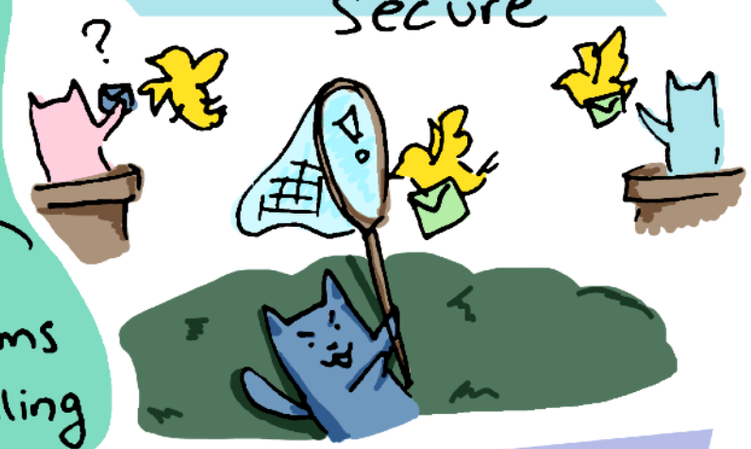
② Latency is ZERO



③ Bandwidth is infinite



④ The network is Secure



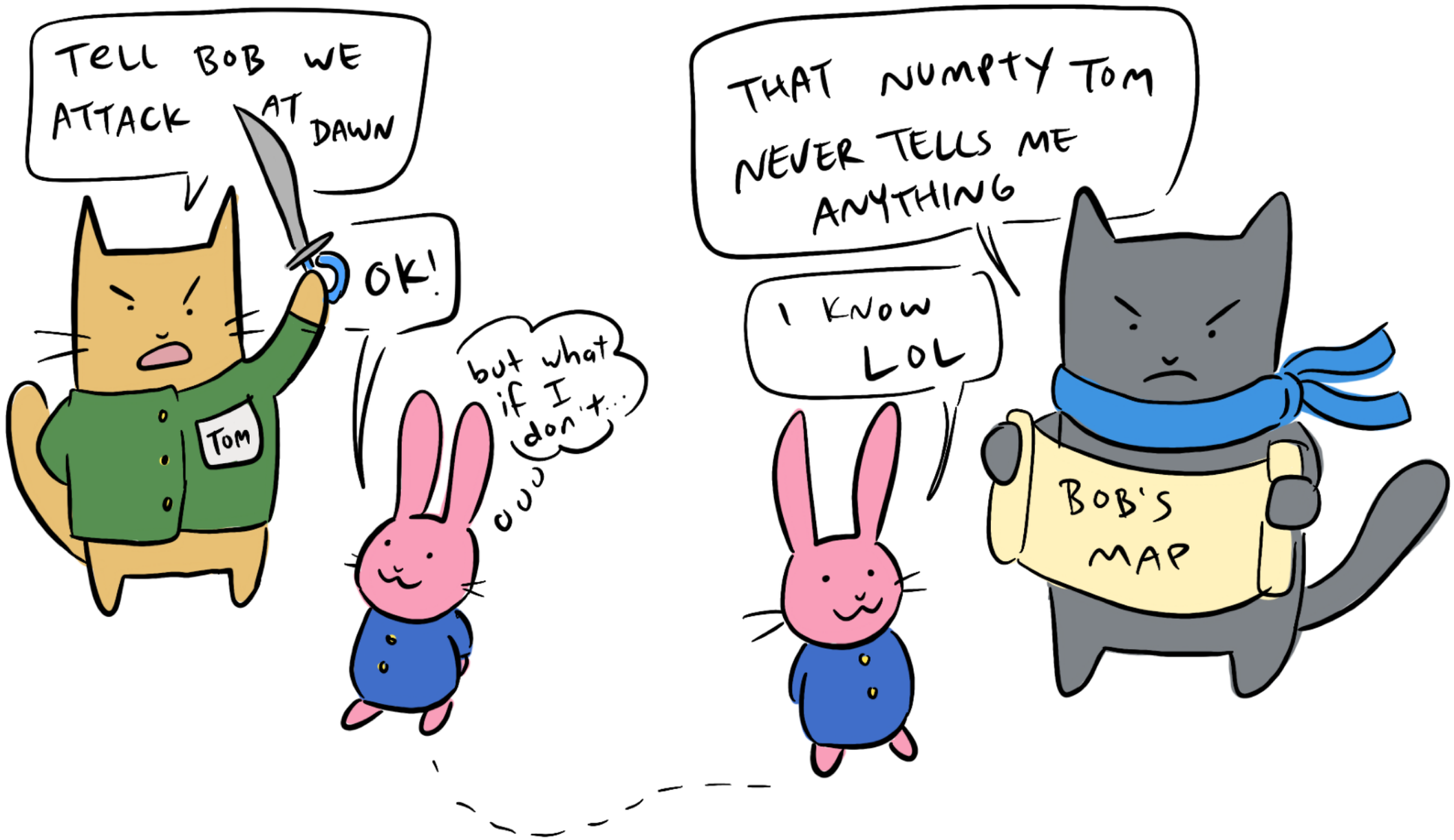
⑤ Topology doesn't change

⑥ There is only one administrator

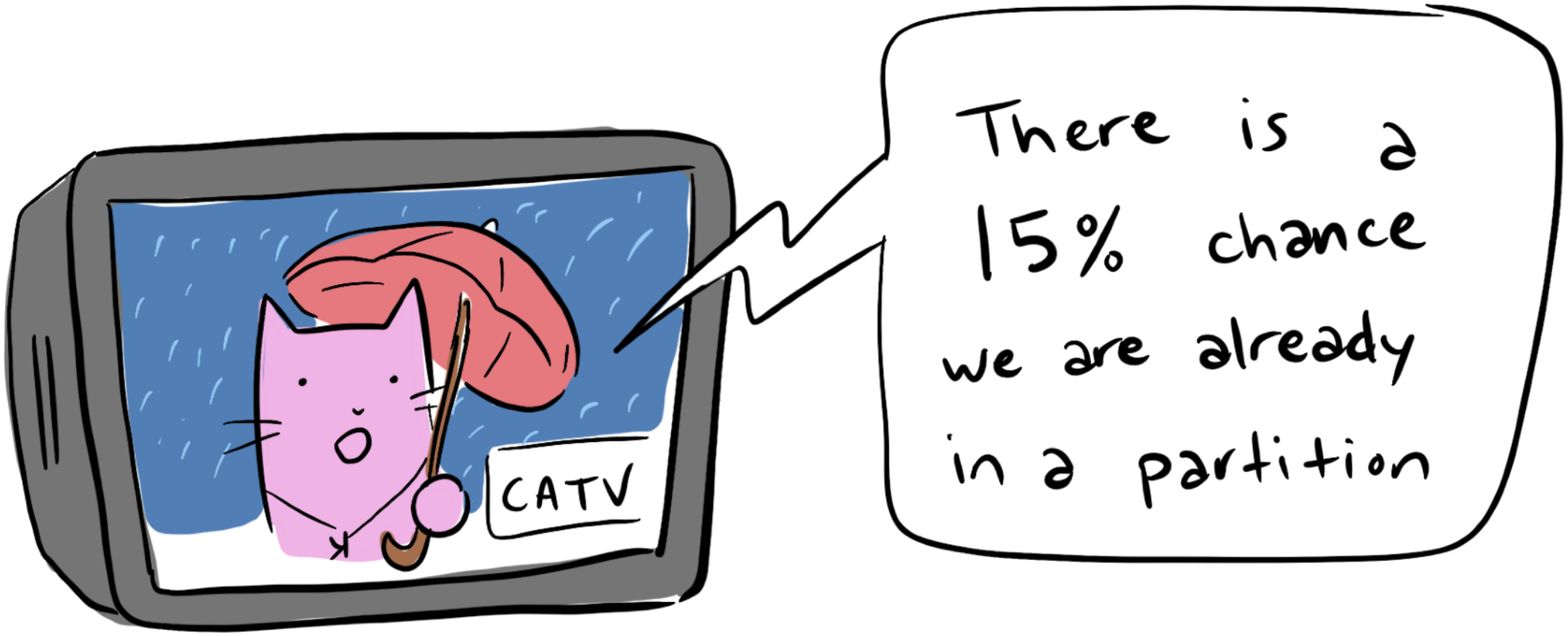
⑦ Transport costs $0

⑧ The network is homogeneous

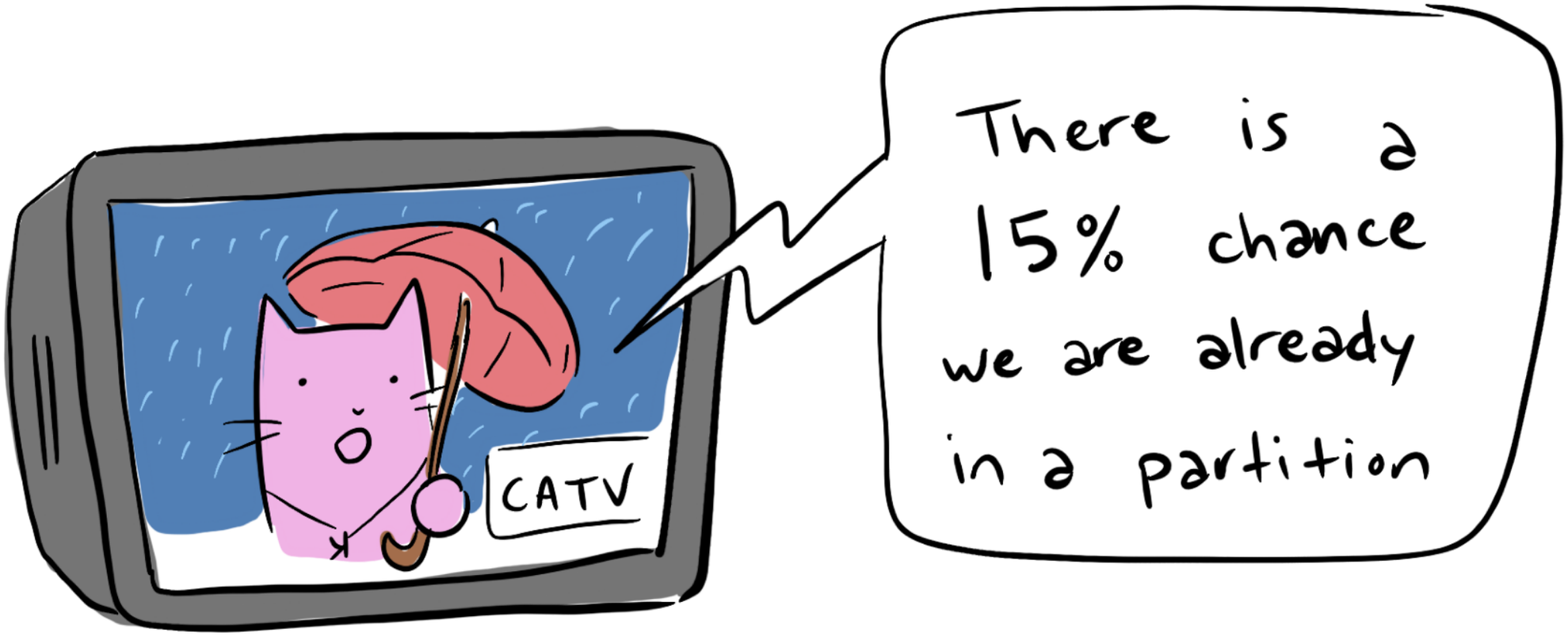
The Byzantine Generals Problem



@deniseyu21



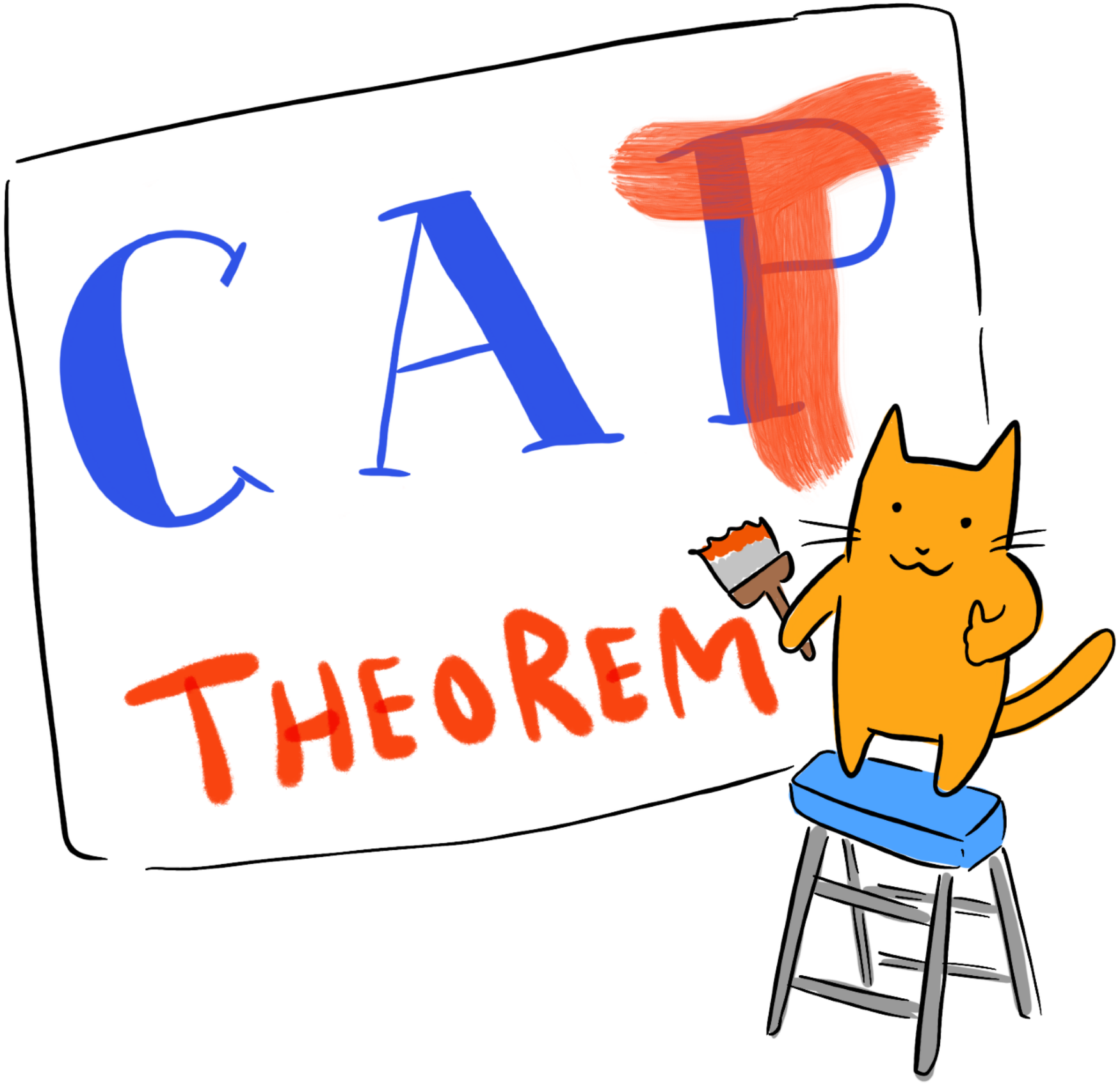
There are lots we can't know. But in distributed computing, we can know one thing:



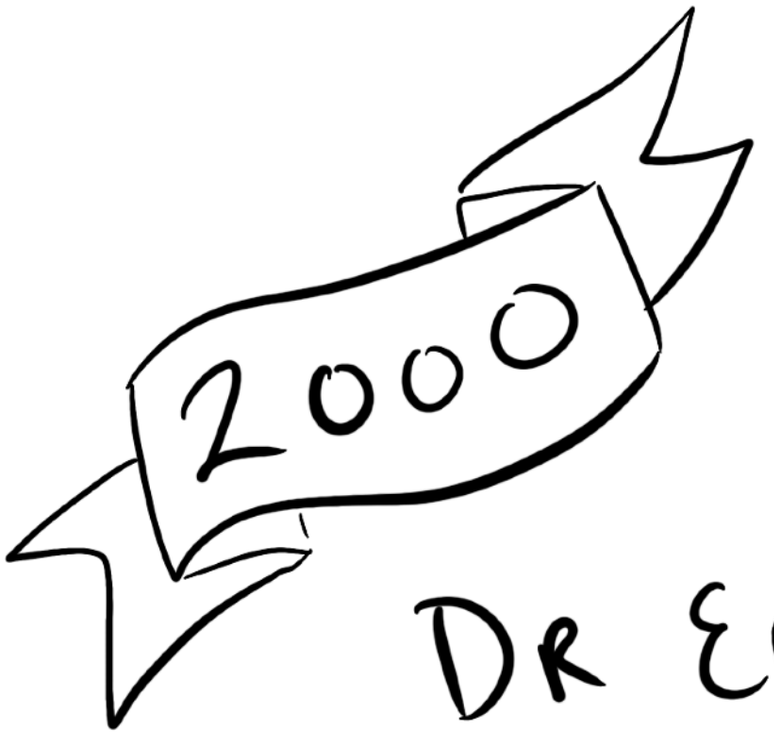
There are lots we can't know. But in distributed computing, we can know one thing:

Shit's gonna fail

@deniseyu21



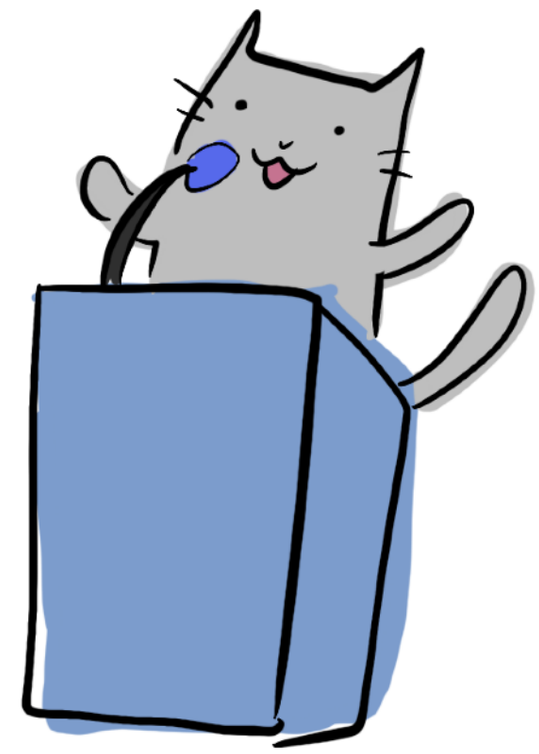
@deniseyu21



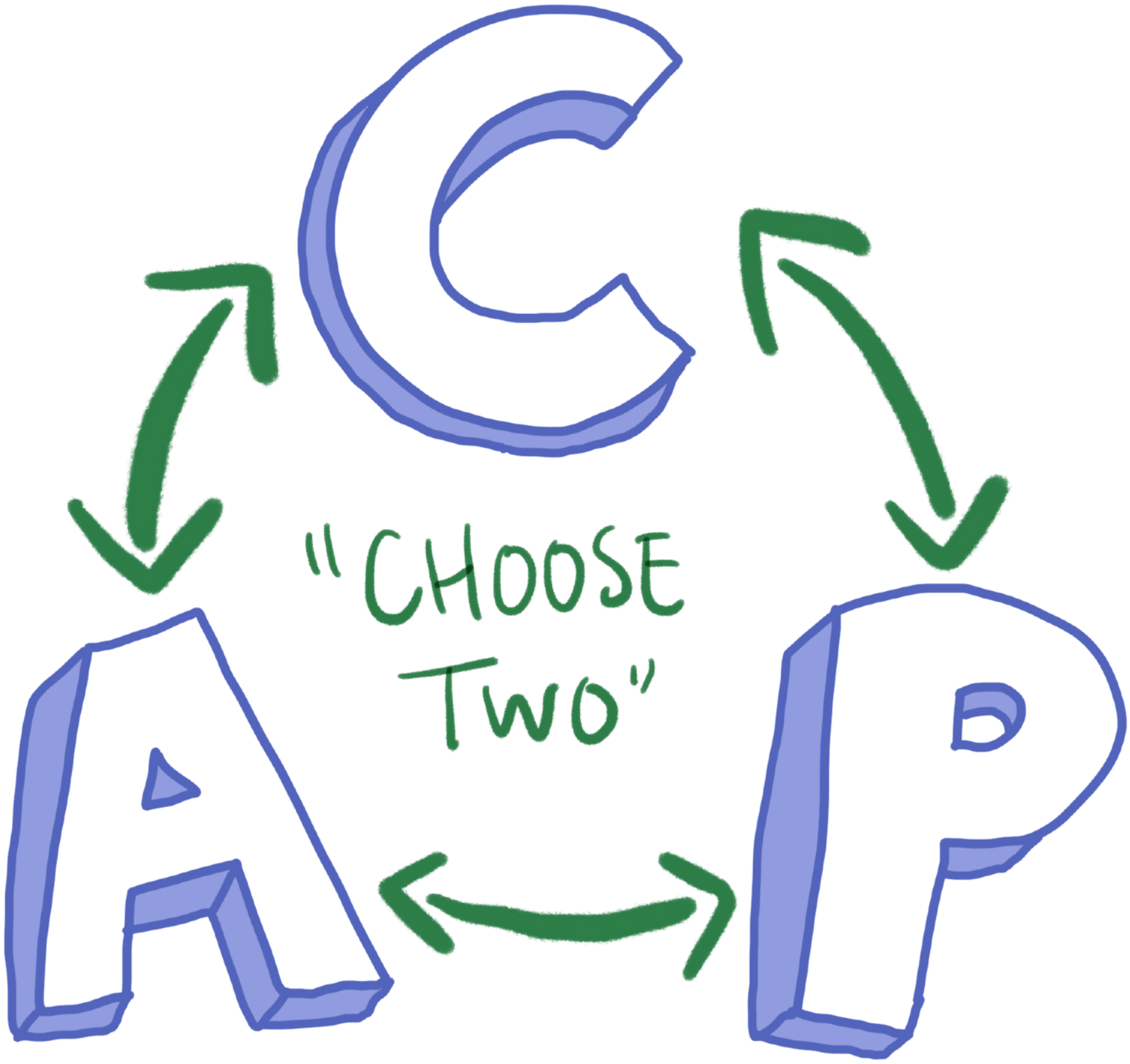
Dr Eric Brewer

"Towards Robust Distributed Systems"

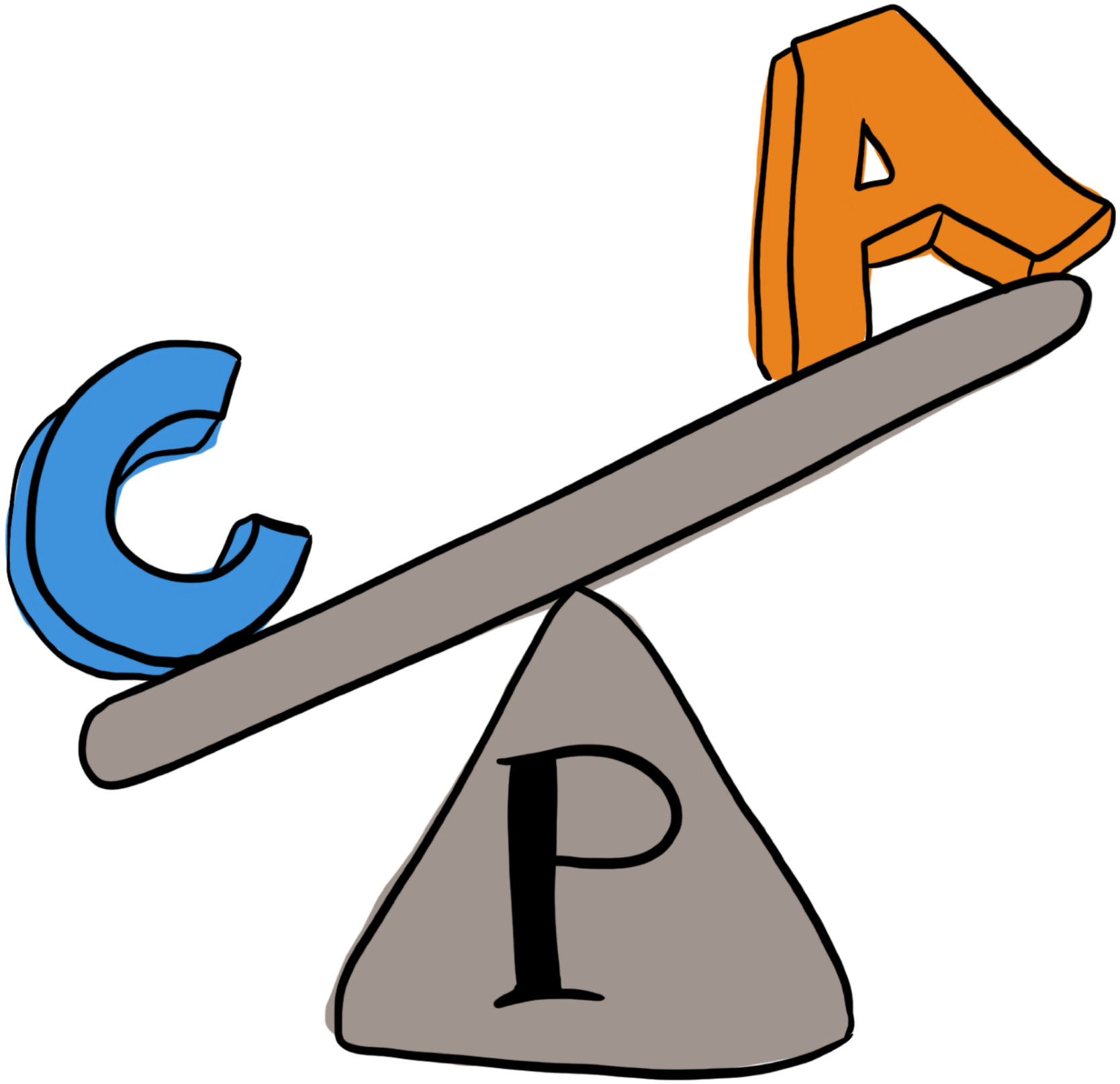
@ Principles of Computing Conf



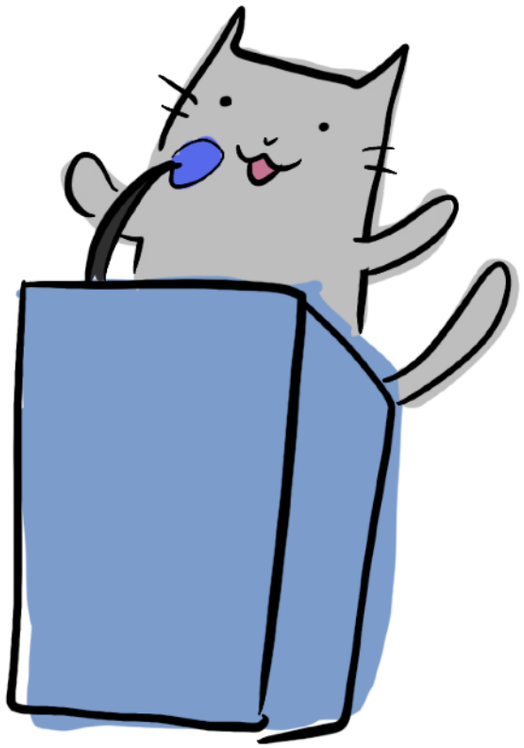
@deniseyu21


C
"CHOOSE TWO"
A
P

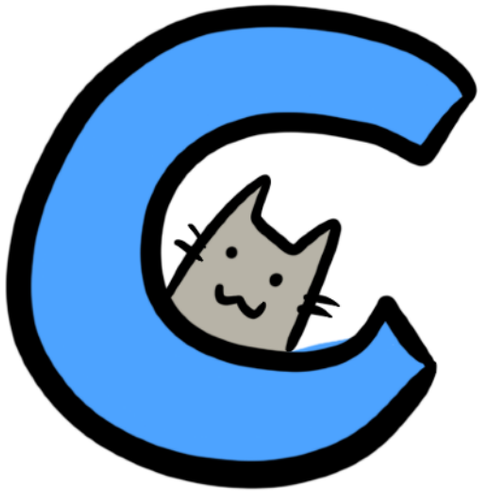

NO!


C
A
P

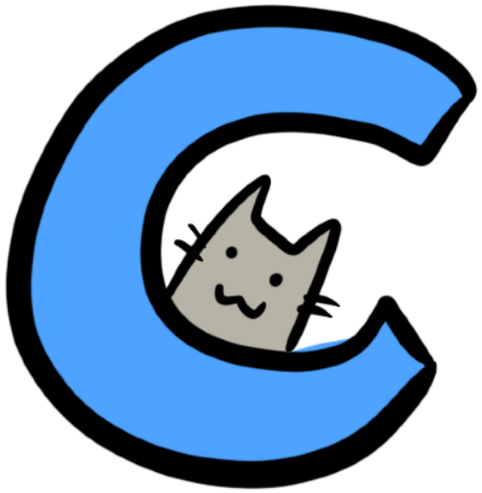
“Does choosing consistency and availability (CA) as the "2 of 3" make sense? As some researchers correctly point out, exactly what it means to forfeit P is unclear.”



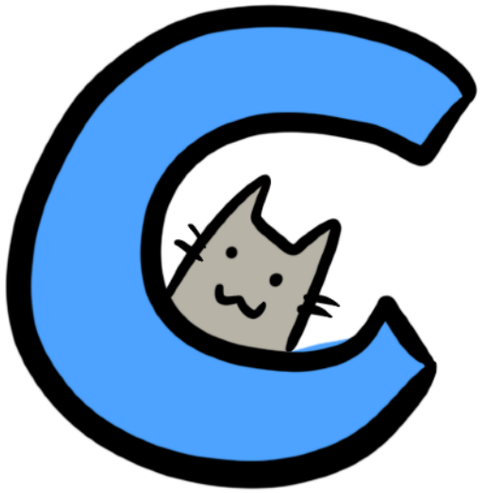
Eric Brewer, *CAP Twelve Years Later: How the 'Rules' Have Changed*



C IS FOR



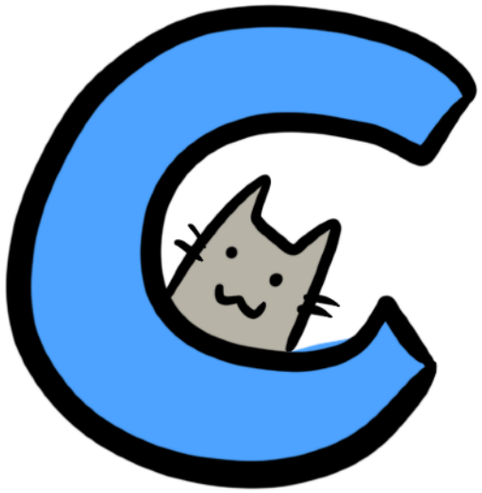
C IS FOR LINEARIZABILITY



C IS FOR LINEARIZABILITY

t_0  cat.state = 'hungry'

t_1  cat.state = 'full'

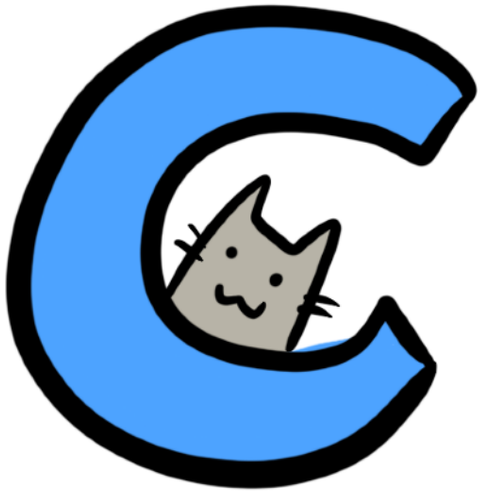


C IS FOR LINEARIZABILITY

t_0  cat.state = 'hungry'

t_1  cat.state = 'full'

<u>All</u> nodes must have t_1 if anyone showed t_1



C IS FOR LINEARIZABILITY

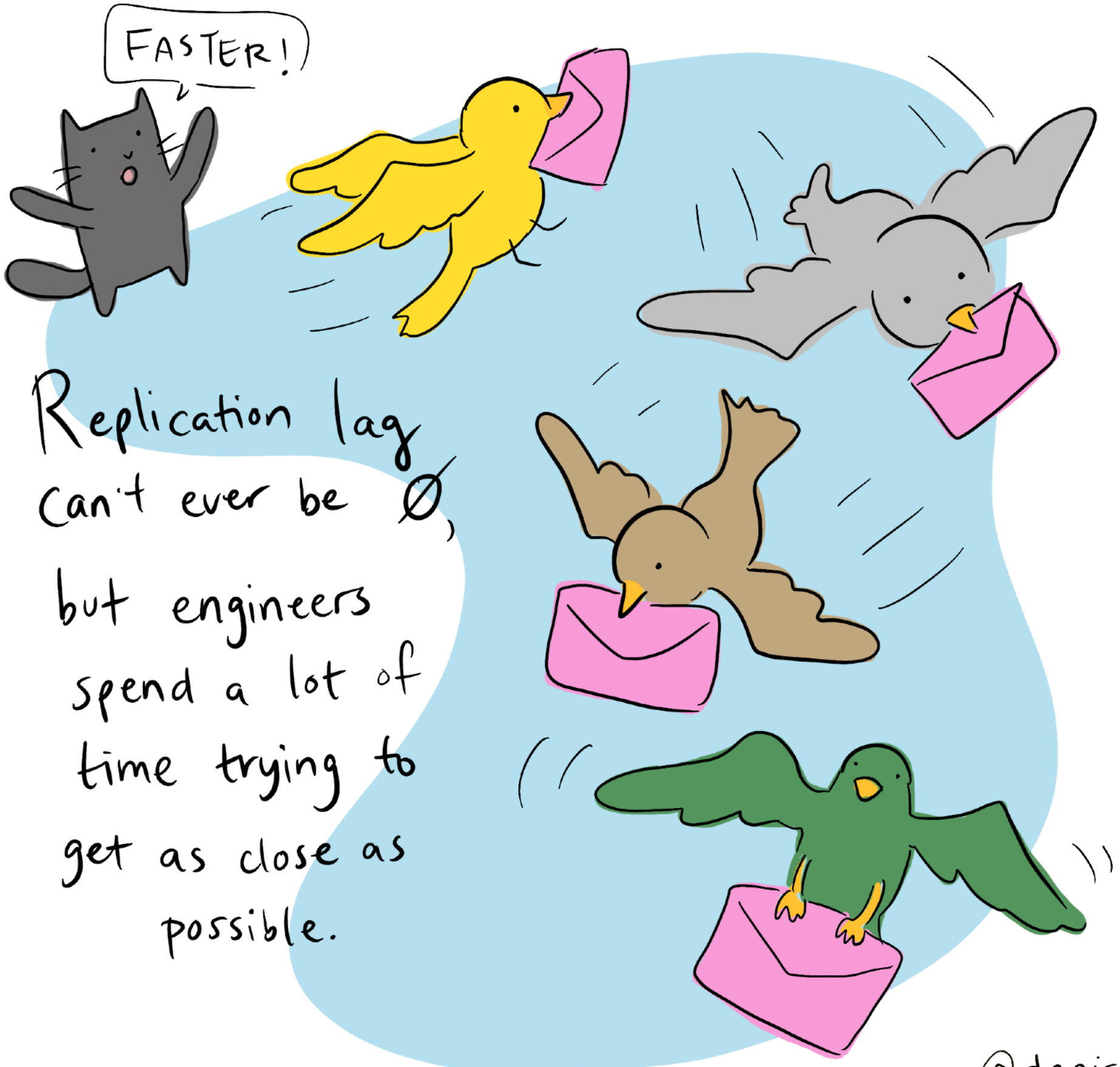
t_0  cat.state = 'hungry'

t_1  cat.state = 'full'

All nodes must have t_1 if anyone showed t_1

This is really hard! Instant & universal replication.

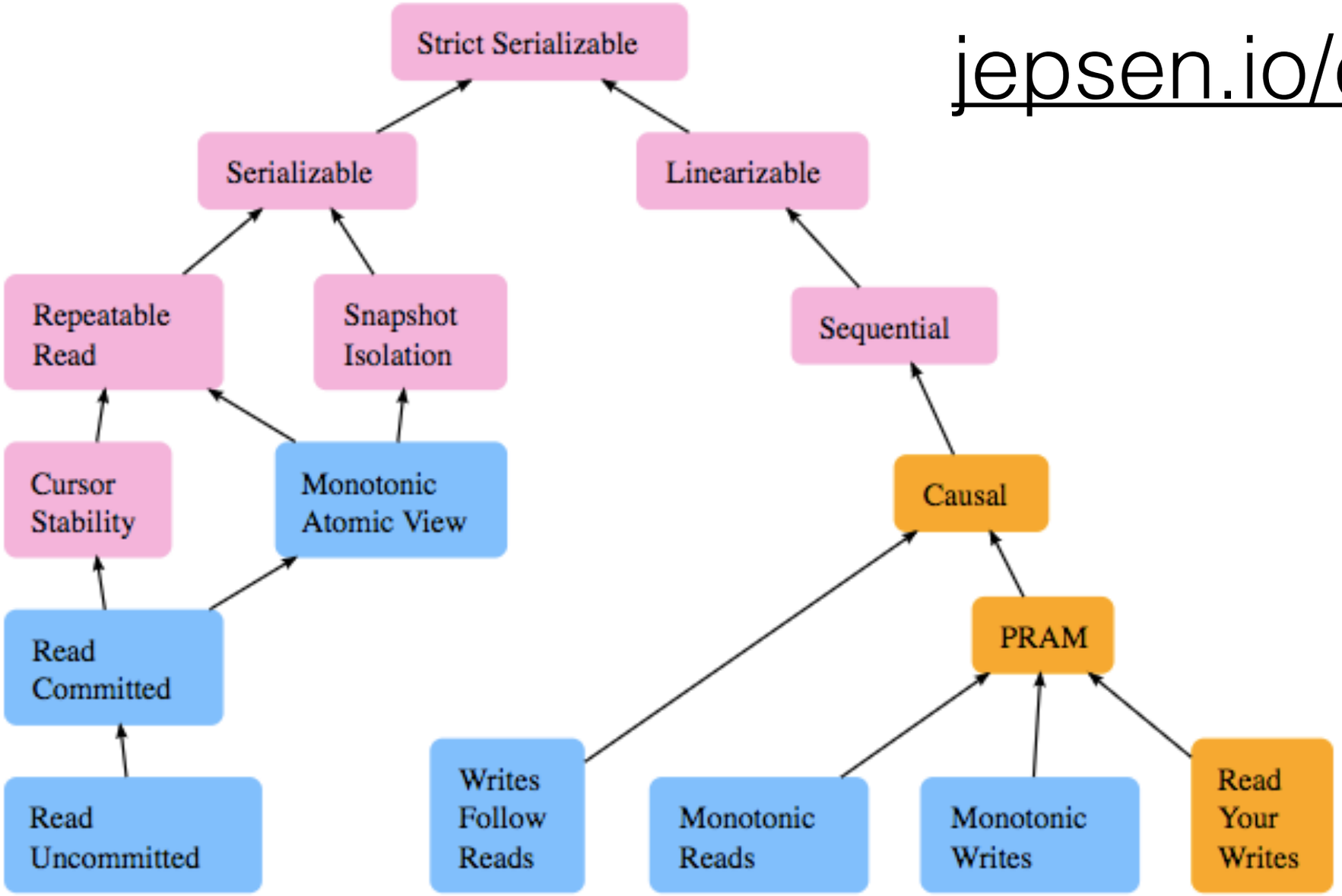
@deniseyu21



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

@deniseyu21

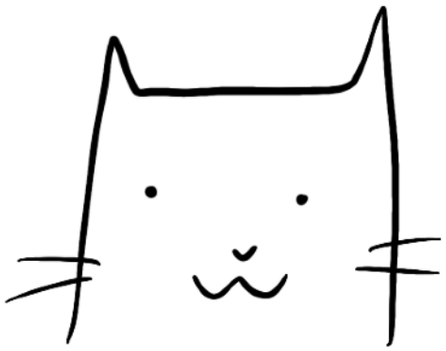
jepsen.io/consistency



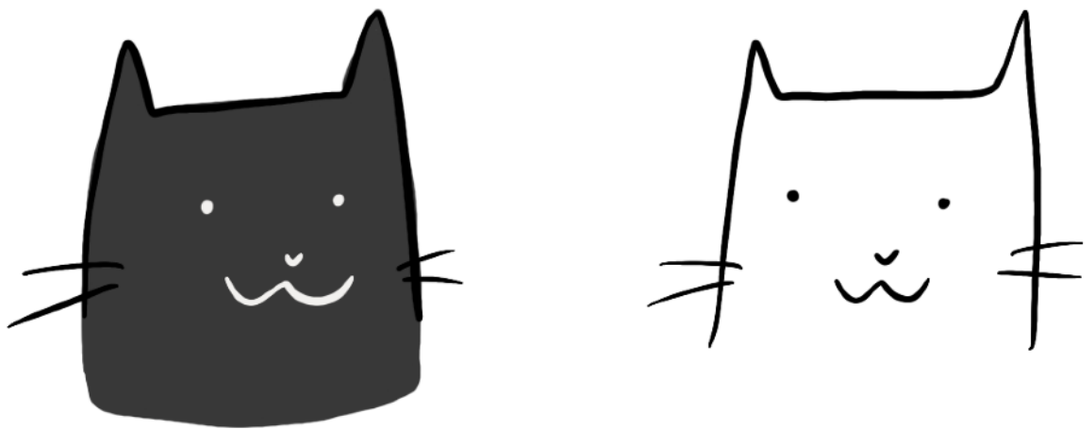
Legend

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



A IS FOR AVAILABILITY



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

Because LATENCY

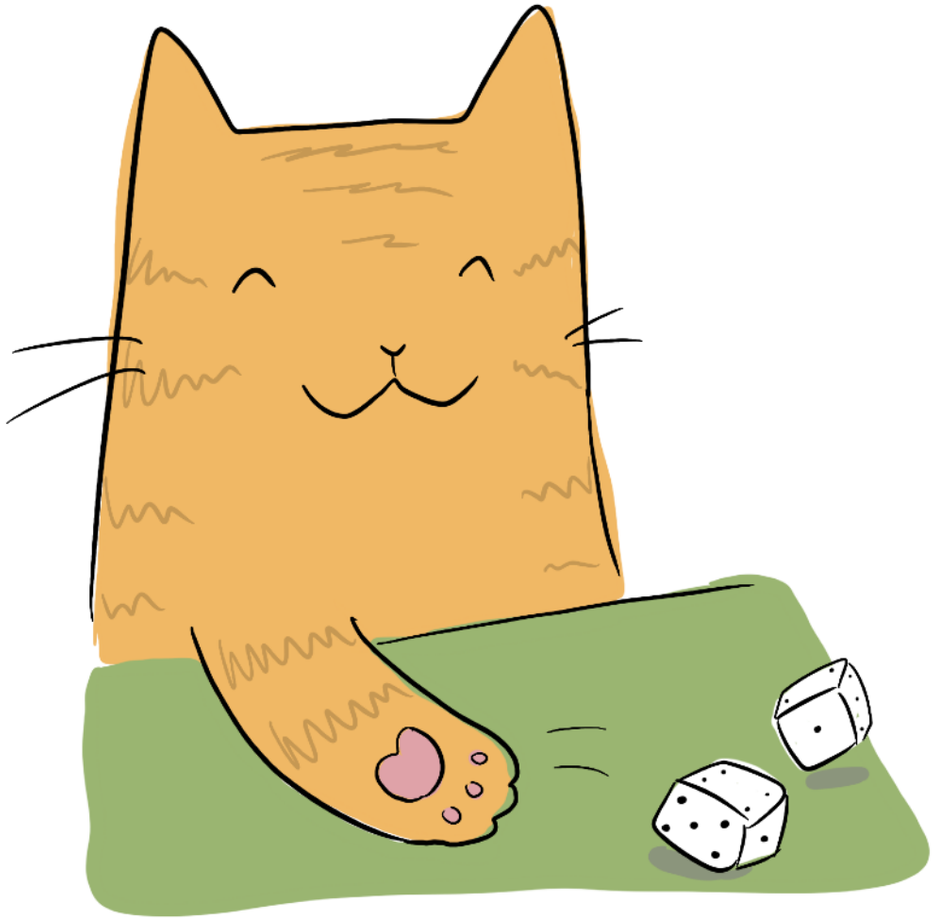
@deniseyu21

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 <u>timeout</u> limit is a very scientific process



yu21

P 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 <u>no</u> way to know what's happening on the other side.

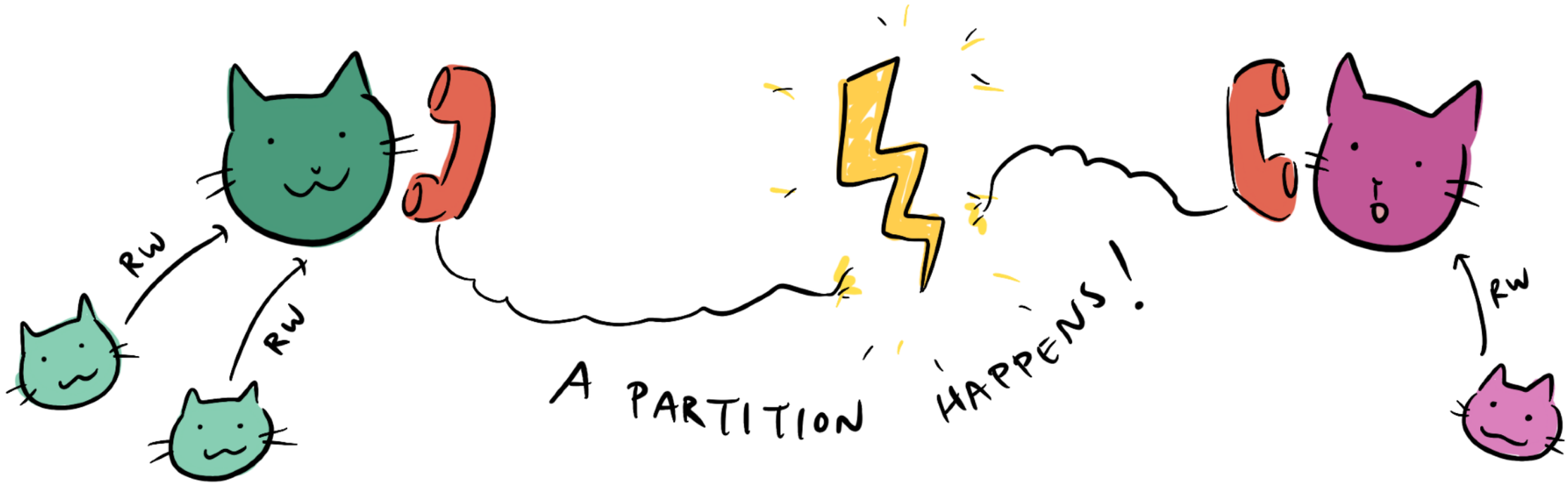
@deniseyu21

C onsistency *

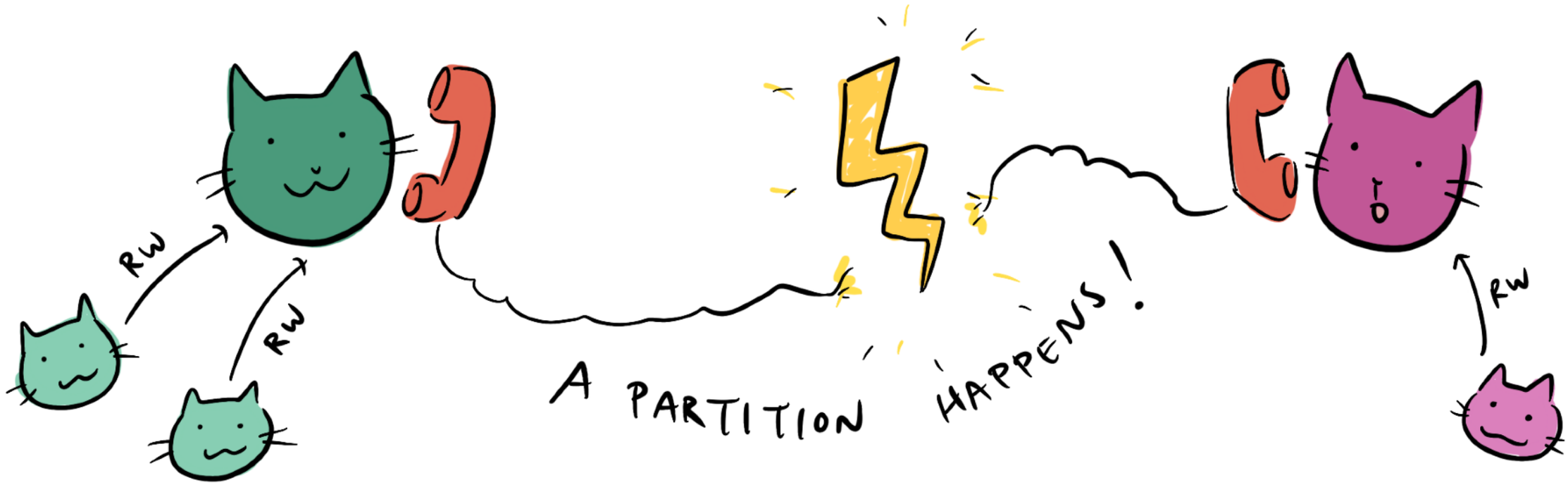
A vailability

P artition tolerance

PROOF OF CAP THEOREM



PROOF OF CAP THEOREM

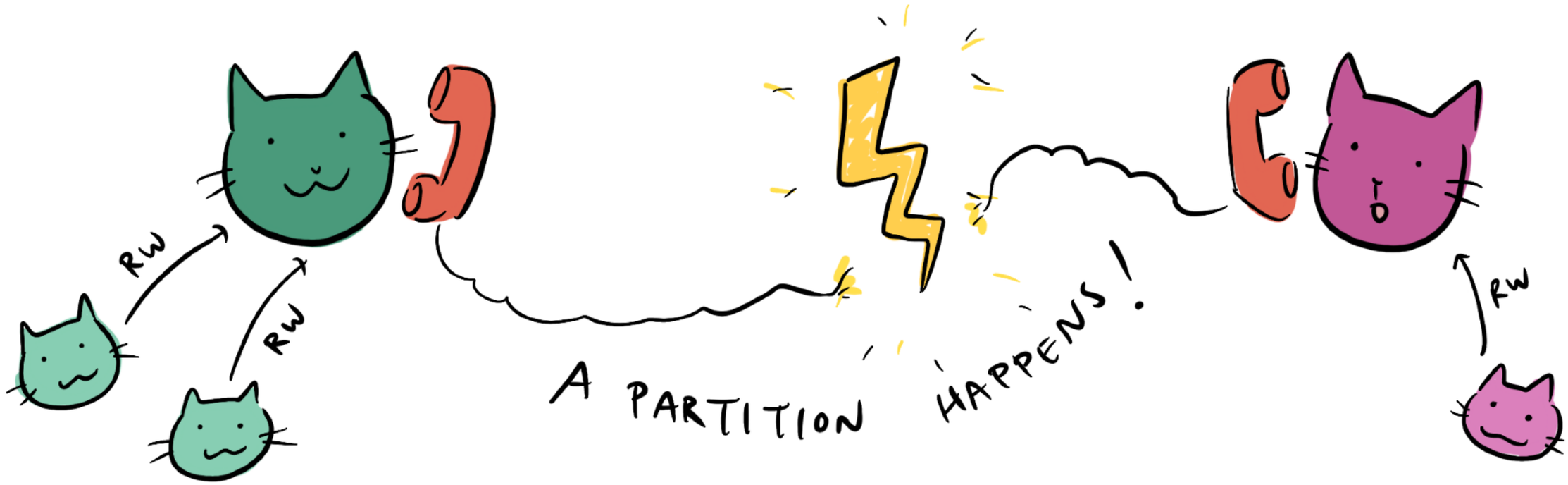


OPTION 1

Let clients keep R/W in both sides of split

✗ LINEARIZABILITY

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

Network partitions
are inevitable.

How inevitable?

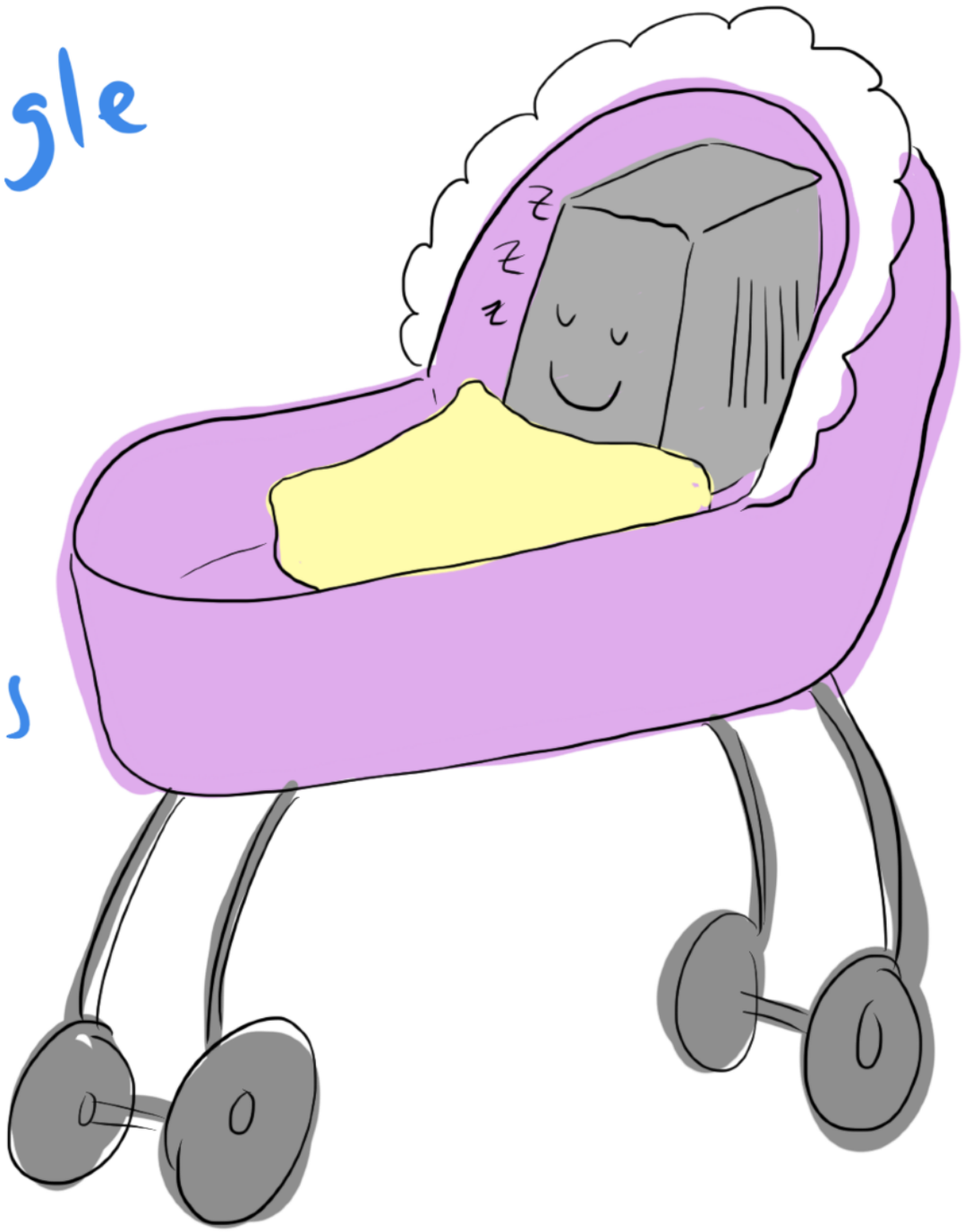
In the first year of a Google cluster's life, it will experience

5 rack failures

3 router failures

8 network maintenances

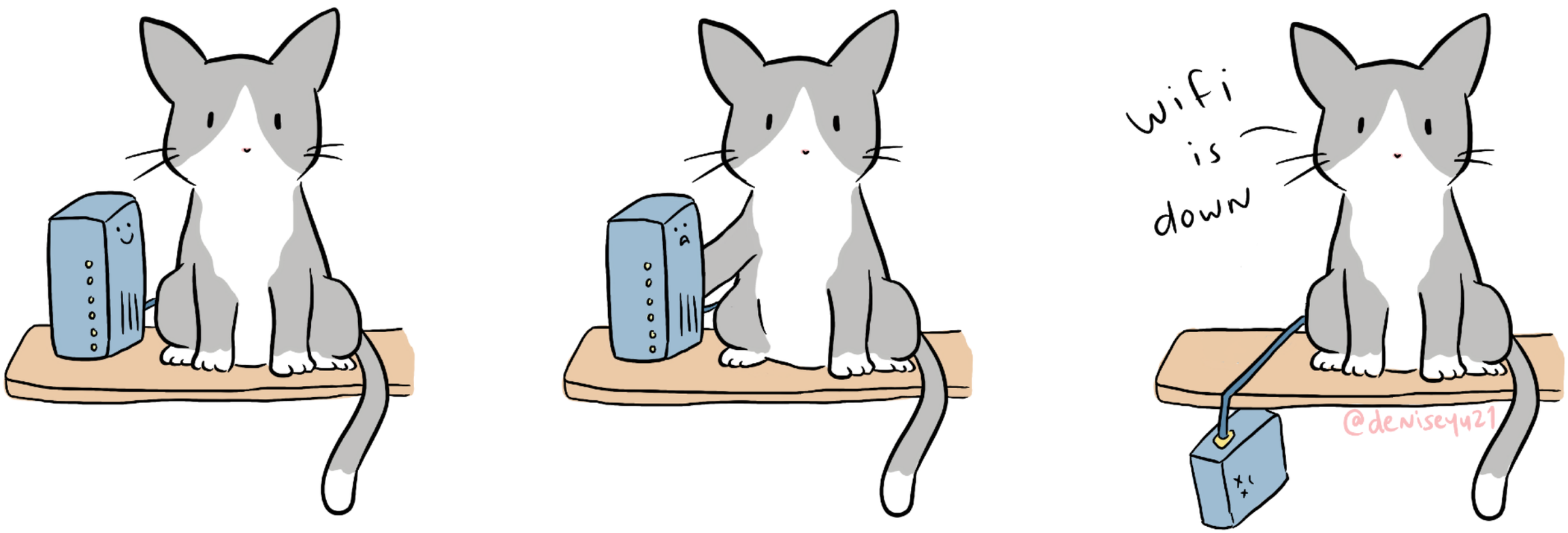
(Jeffrey Dean)



@deniseyu21

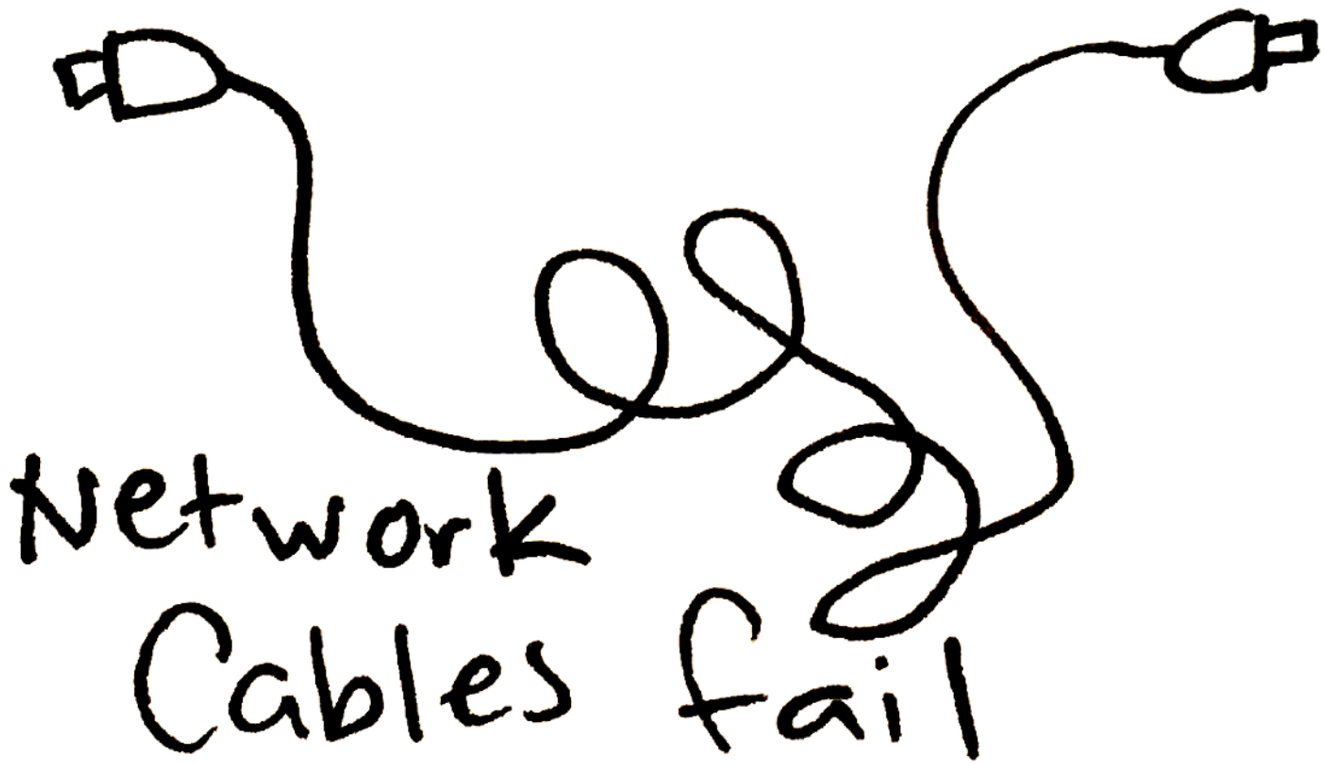
Hardware <u>will</u>
fail

Hardware will fail



@deniseyu21

Hardware <u>will</u> 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

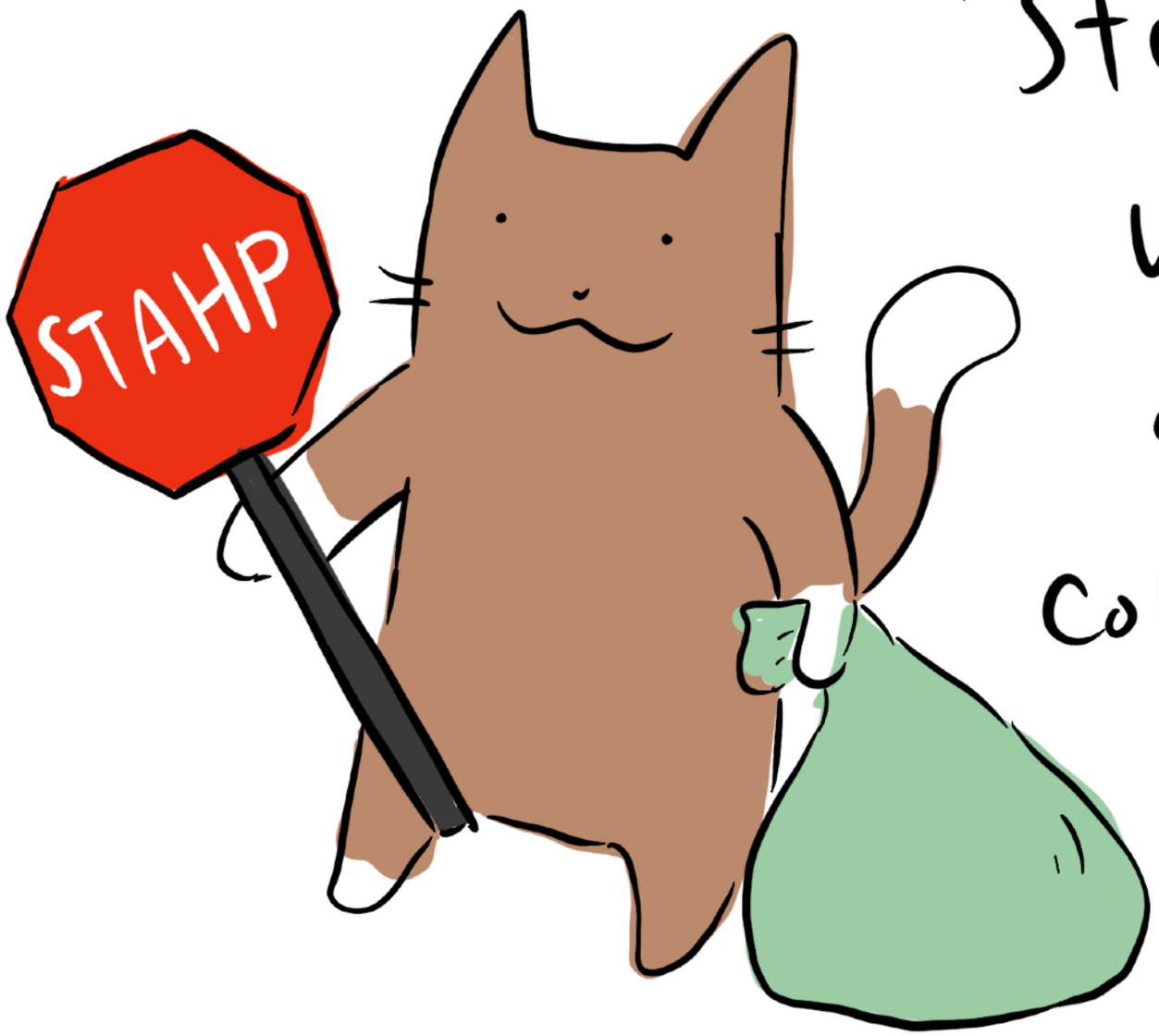
Software will behave weirdly

"Bursty" VMs borrow resources from each other –

the Noisy Neighbor Problem

@deniseyu21

Software will behave weirdly



"Stop the WORLD" garbage collection

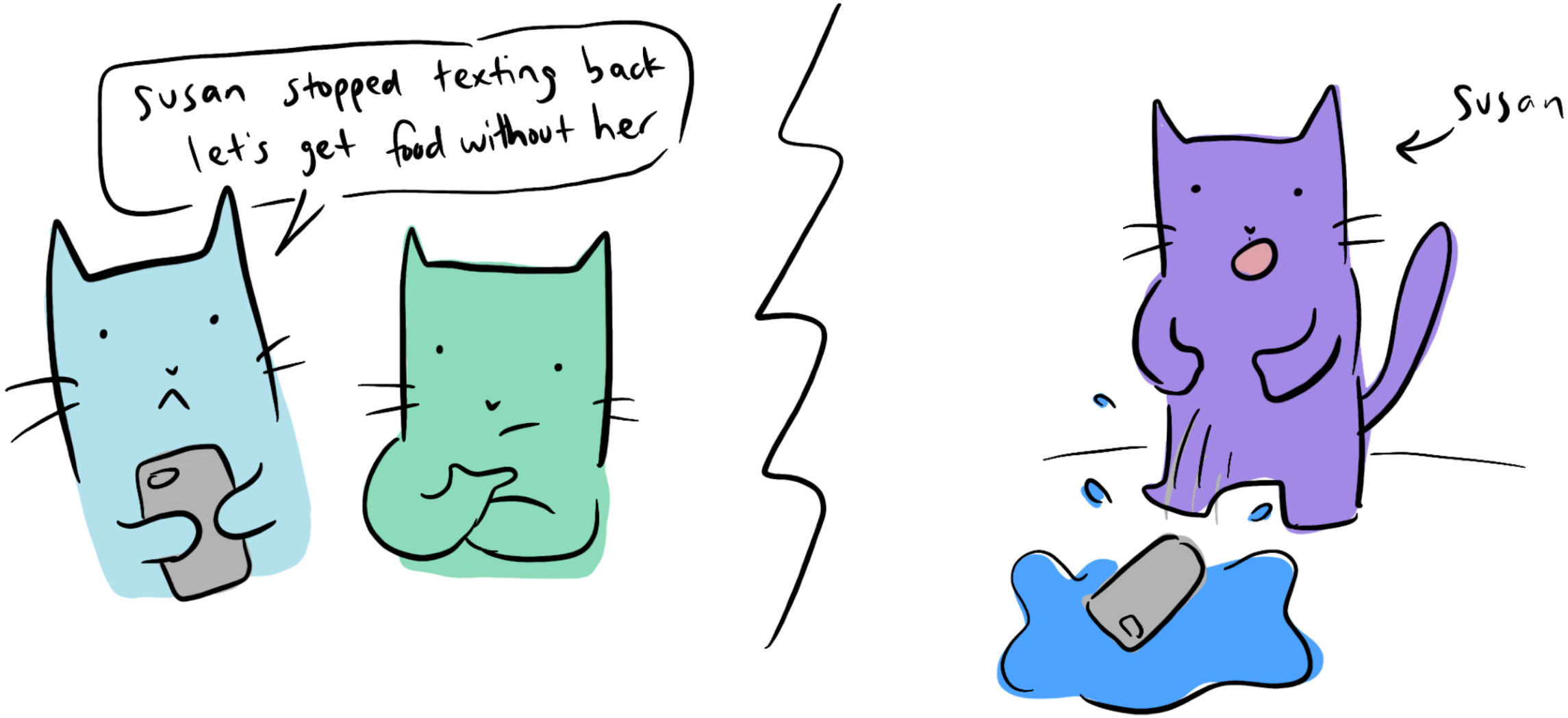
@deniseyu21

Network glitches

@deniseyu21

why does any
of this matter?

Some part of every system is always at risk of failing

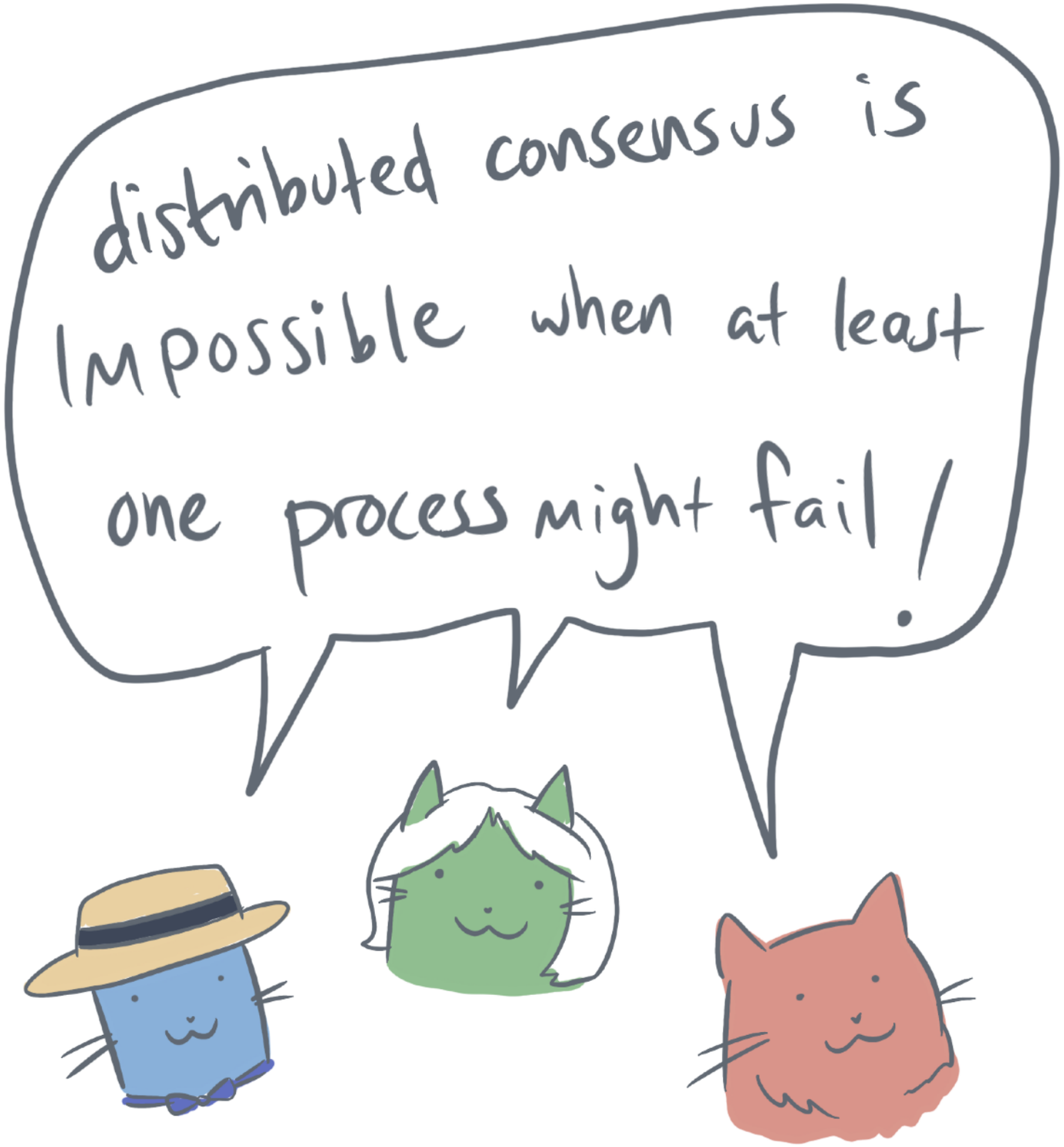


@deniseyu21

Fischer Lynch Paterson

Correctness result

@deniseyu21



“Impossibility of Distributed Consensus with One Faulty Process, 1985.

@deniseyu21

to manage uncertainty, we have mitigation strategies

make rules for how many "yes" votes is enough to proceed

How does RAFT work?

RAFT is a **CONSENSUS ALGORITHM**

a process for getting multiple machines to "agree"!

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

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

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

COPY me!

Ack!

Ack

Followers copy leader

Leader "commits" the new data

FISH :

& followers increment their state counter

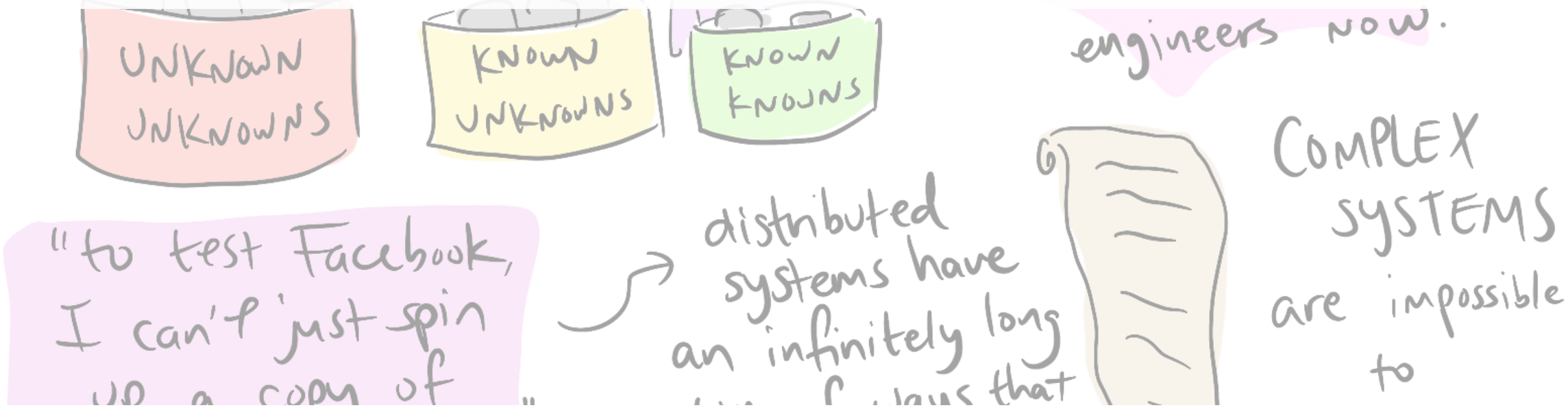
@deniseyu21 with @mt165

what is even
harder than
getting machines
to agree?

OBSERVABILITY

"Systems are getting more complicated, mental models are getting tougher to build, and we're all distributed systems engineers now."

Charity Majors (@mipsytipsy)



humans

how do we manage growing complexity?

understand where

COGNITIVE COMPLEXITY

comes from

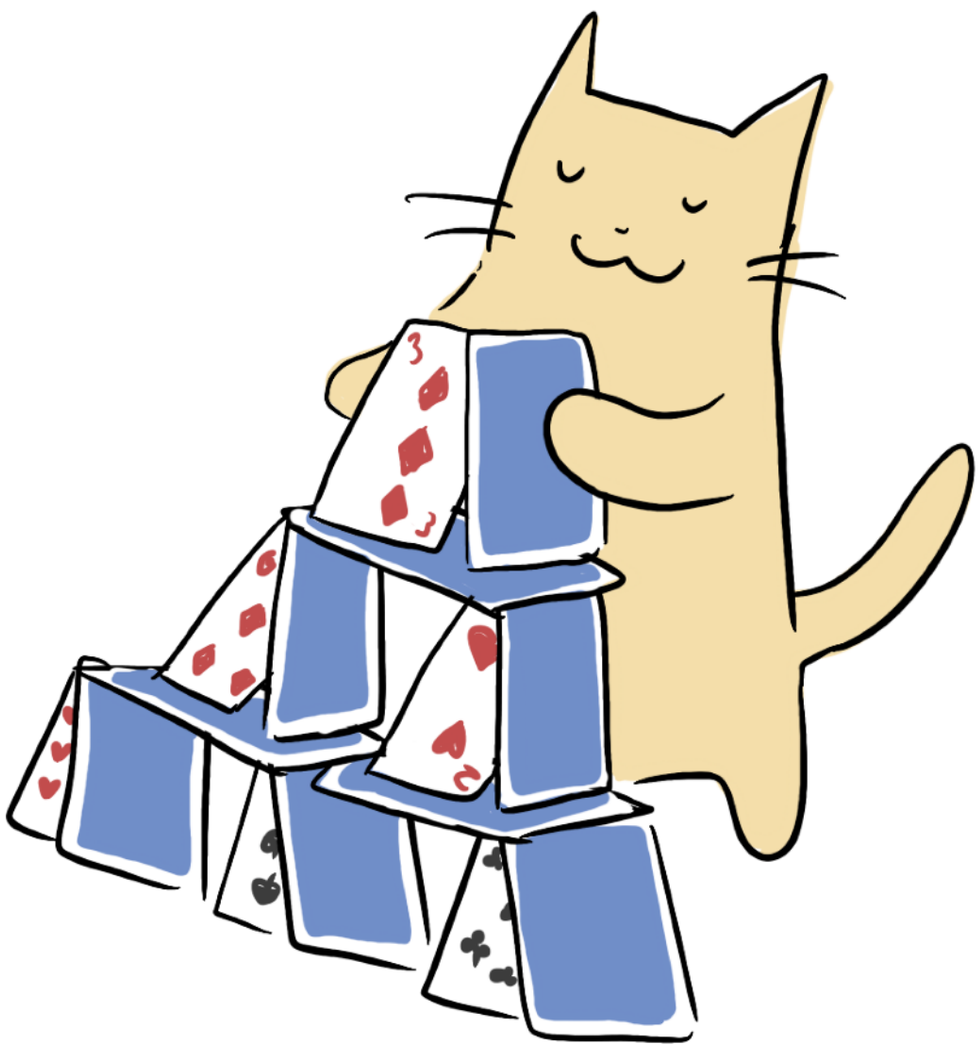
Woods' theorem:

"as the complexity of a system increases, the accuracy of any single agent's own model of that system decreases rapidly."

"Coping With Complexity: the psychology of human behavior in complex systems." Dr. David Woods, 1988.

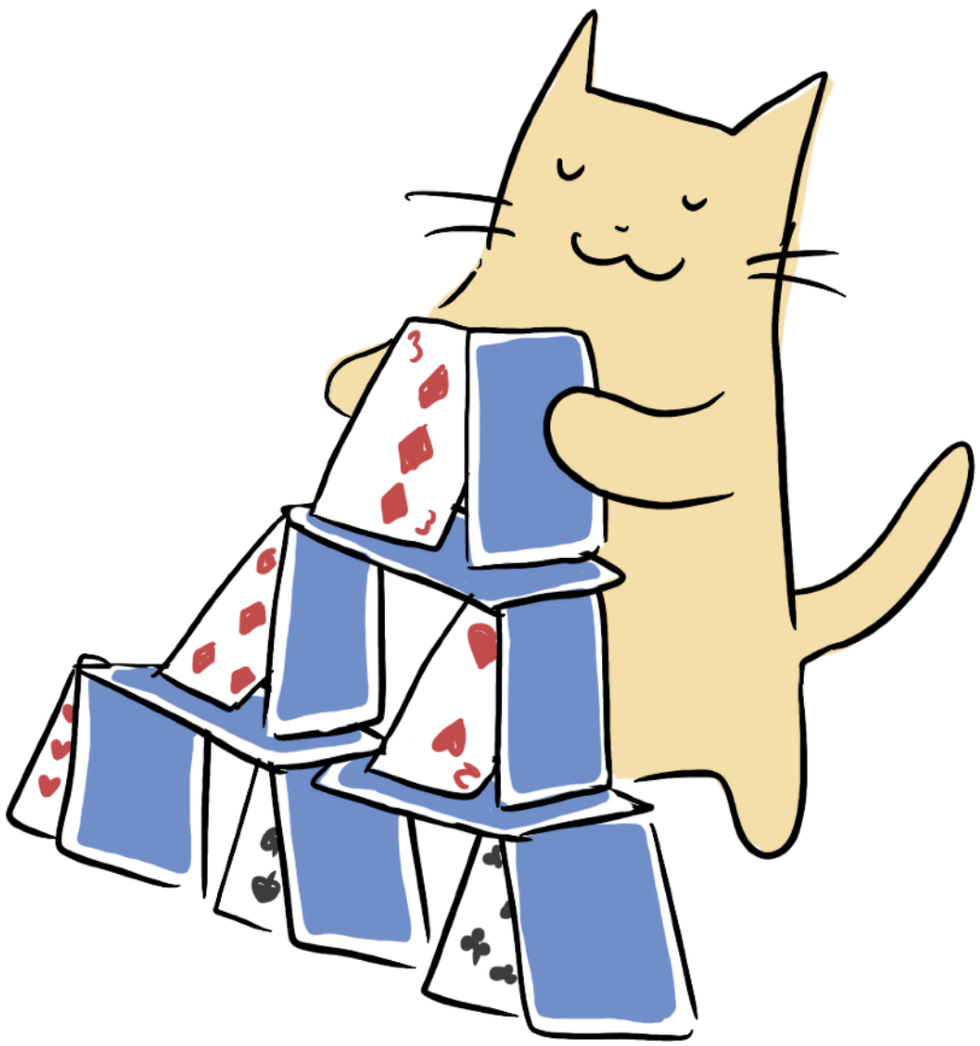
BUILDING MENTAL MODELS

in theory:



BUILDING MENTAL MODELS

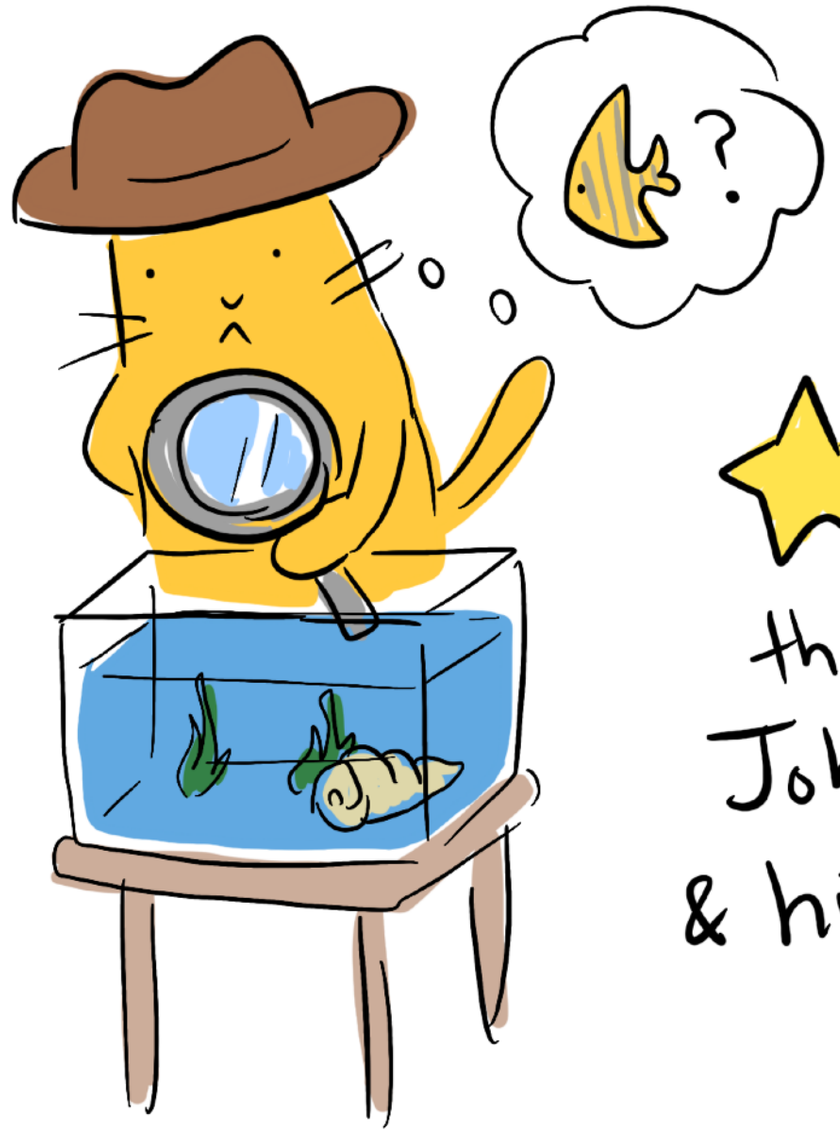
in theory:



in practice:

@deniseyu21

INCIDENT ANALYSIS



This is the work of John Alls & his team

is particularly great for mental model calibration

@deniseyu21

Art created for zine.incidentlabs.io

Blameless discussions optimized for learning

@deniseyu21

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

Unintuitive design?

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

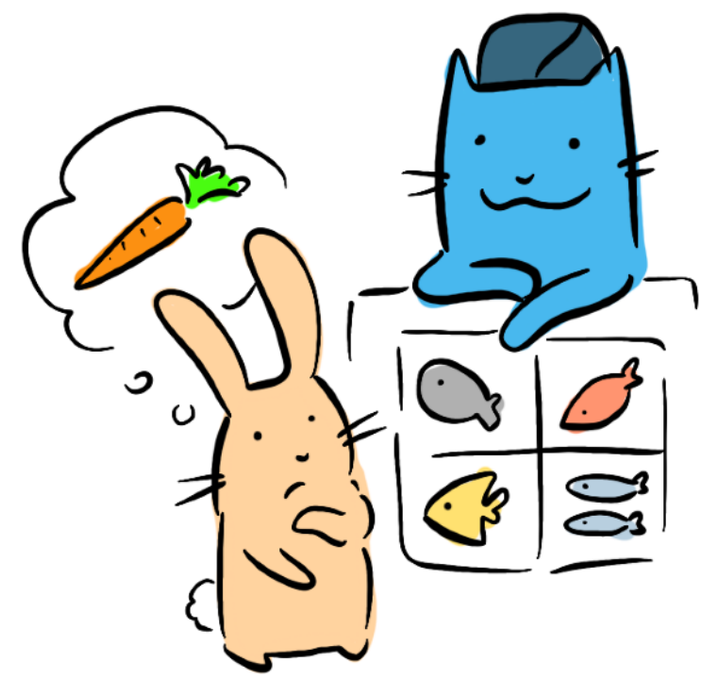
Alert fatigue?

Unintuitive design?

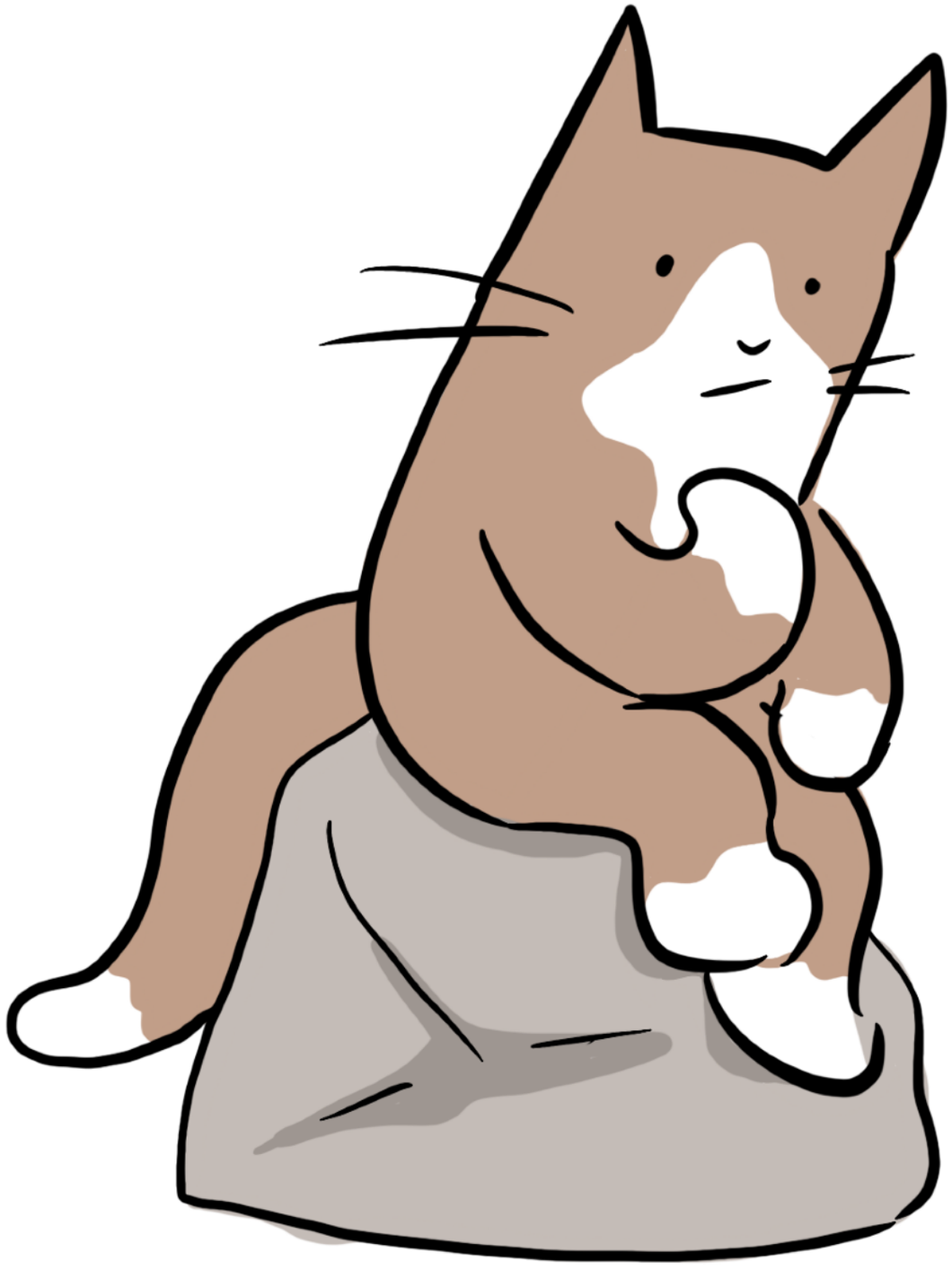
☐ check if you do not not not not not not not wish to receive emails

Alert fatigue?

Not understanding the users' assumptions and needs?

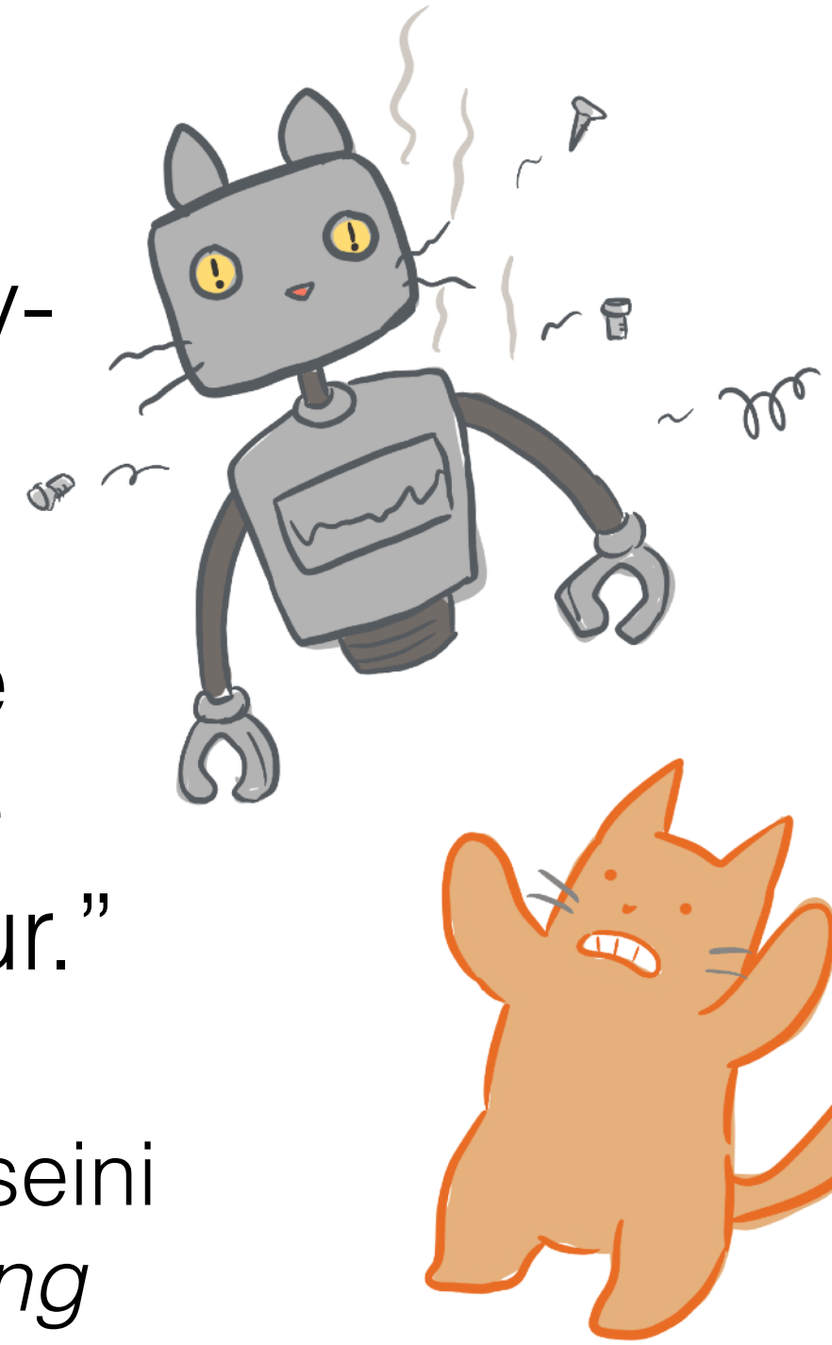


@deniseyu21



Does the
root cause
even exist?

“The more we depend on technology and push it to its limits, the more we need highly-skilled, well-trained, well-practiced people to make systems resilient, acting as the last line of defence against the failures that will inevitably occur.”



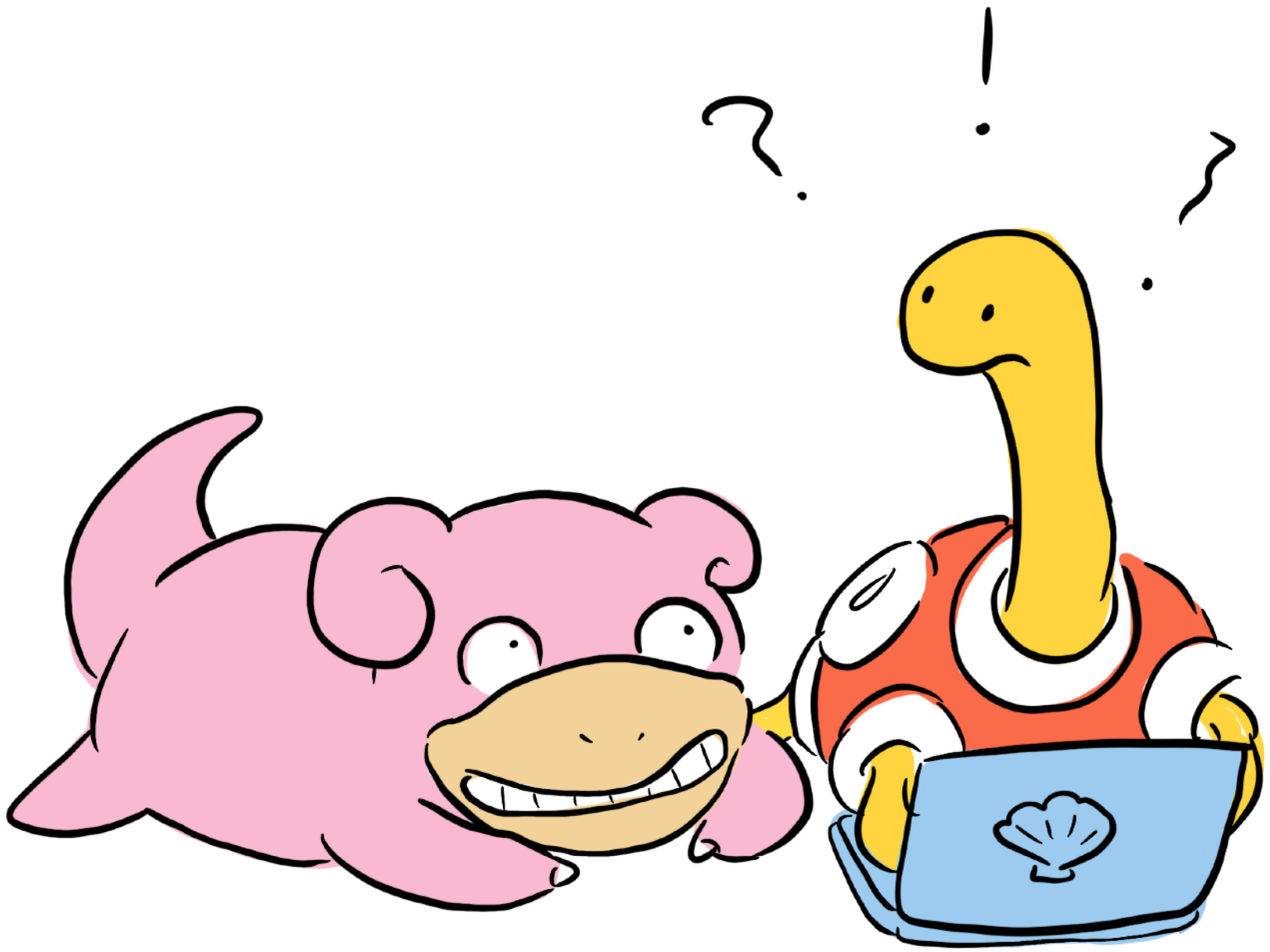
Baxter, Rooksby, Wong, Khajeh-Hosseini
“The Ironies of Automation… still going strong at 30?” (2012)

we borrow against

inherent complexity

but we can learn

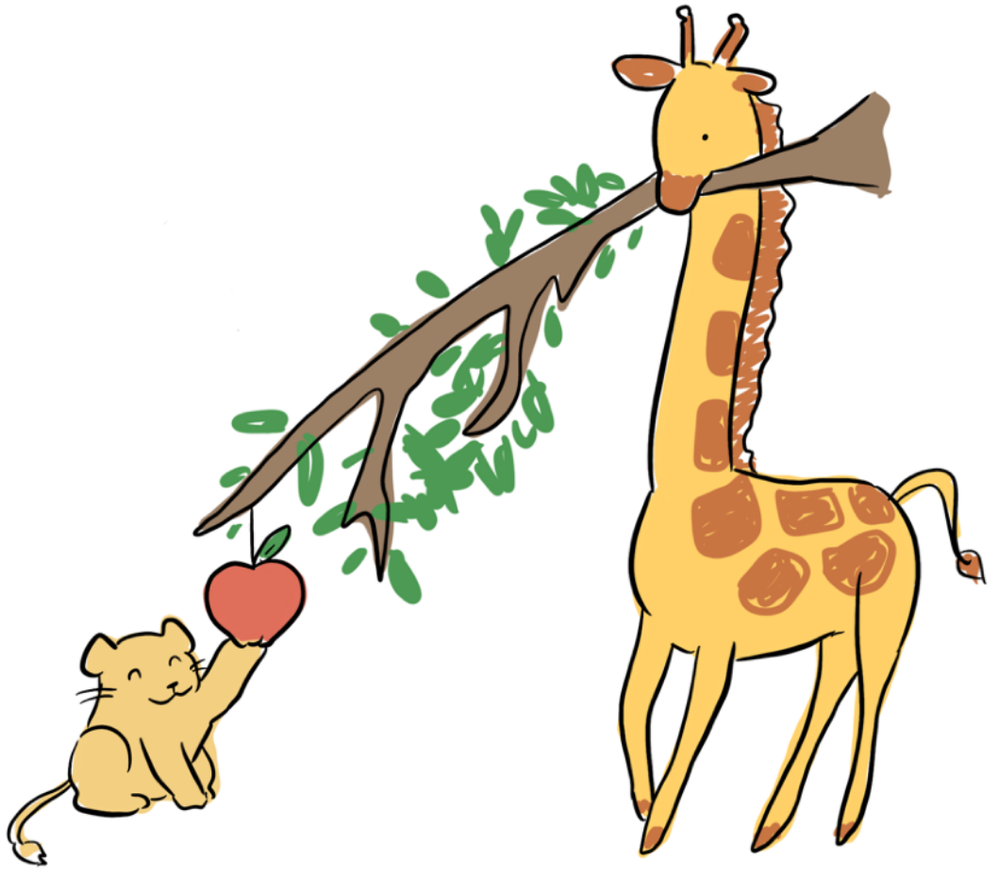
and adapt



Challenge yourself to empathize with each user

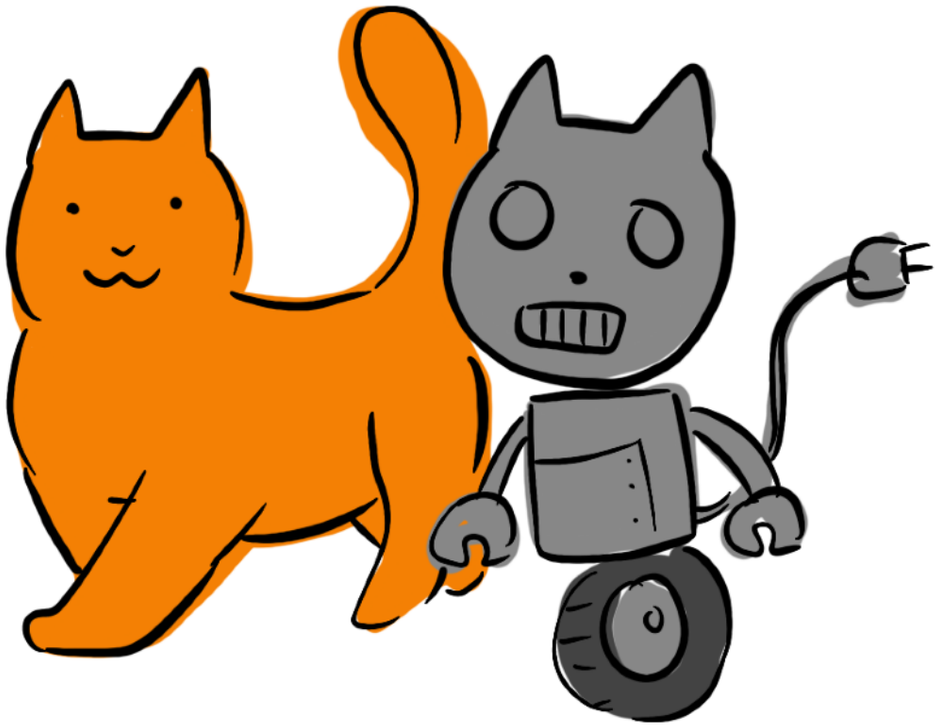
@deniseyu21

Our design choices should make life simpler for the humans operating our systems



@deniseyu21

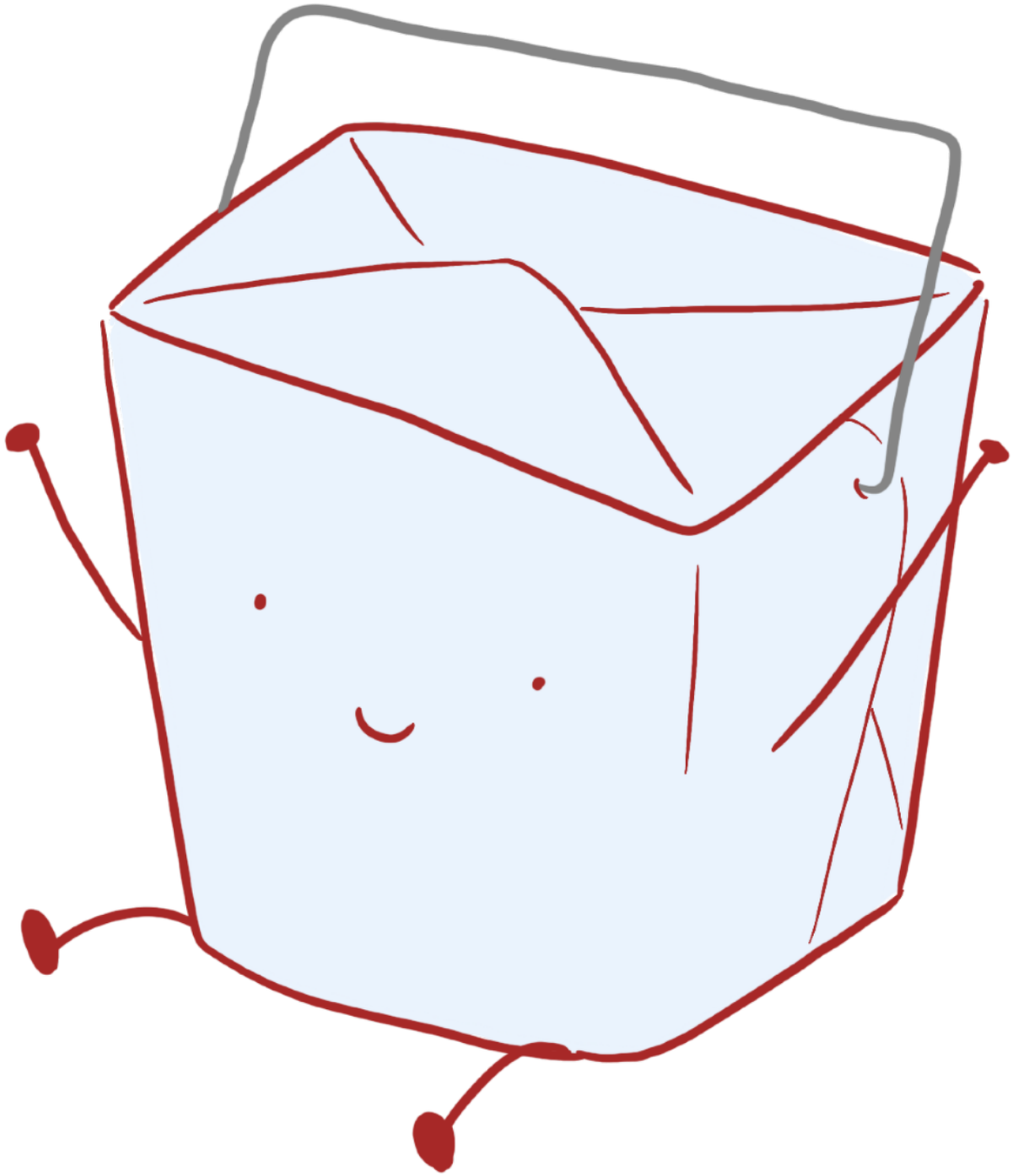
we owe it to our end users &
our teams to understand
& design for the whole system



including the fleshy human parts.

@deniseyu21

Thanks!



@deniseyu21