

Cloud native is about culture, not containers



Dr Holly Cummins
Worldwide Development Lead
IBM **Cloud** Garage

by the way ...

the IBM Cloud Garage is hiring :)



what **is** cloud native?



Daniel Bryant

@danielbryantuk

Following



I've gotta hand it to [@bibryam](#), he's got a great way of framing things... :-)

Microservices
(smart endpoints, dumb pipes)

Bilgin Ibryam @bibryam
ESB -> Microservices -> CloudNative

8:42 AM - 7 Aug 2018

2 Retweets 7 Likes

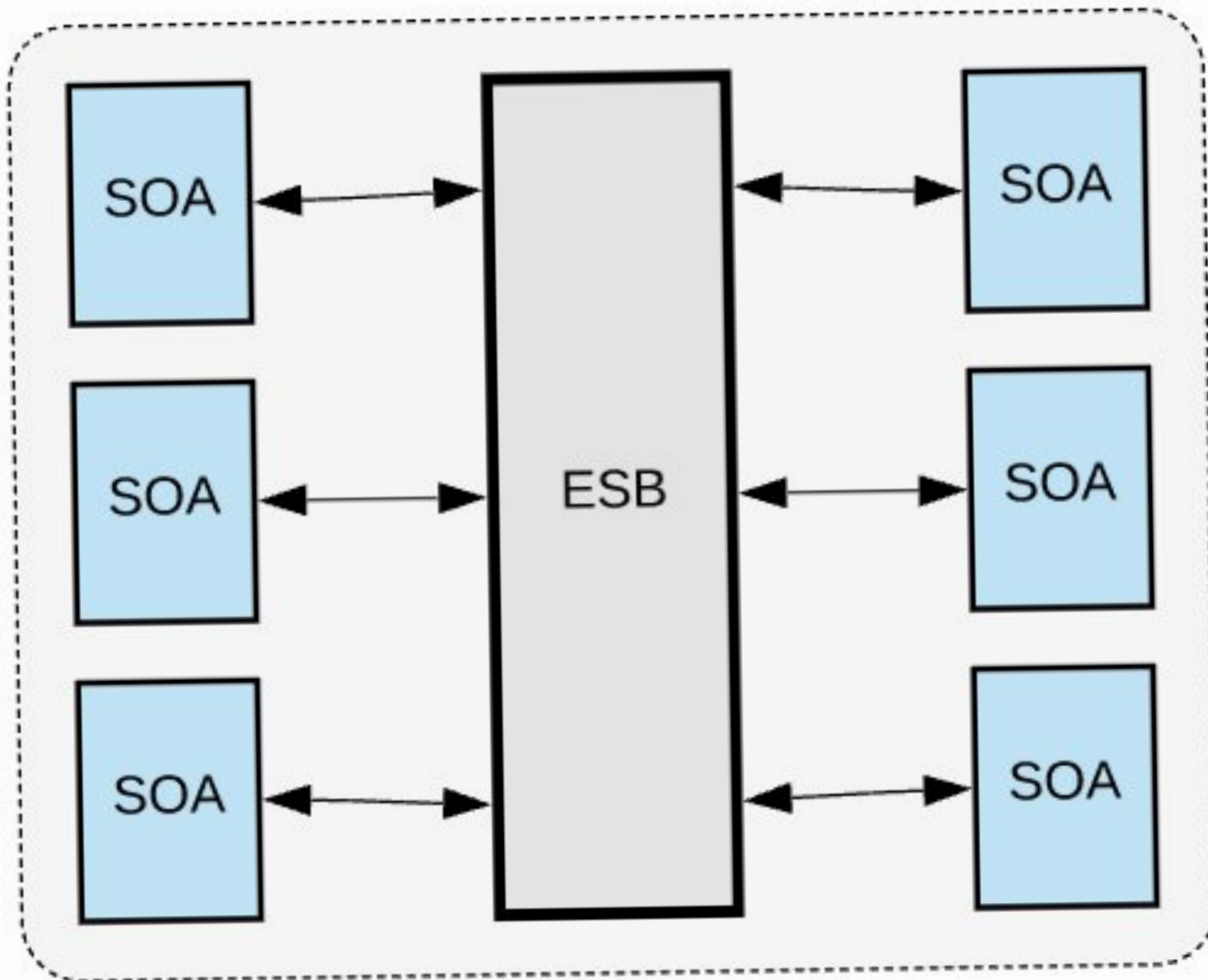




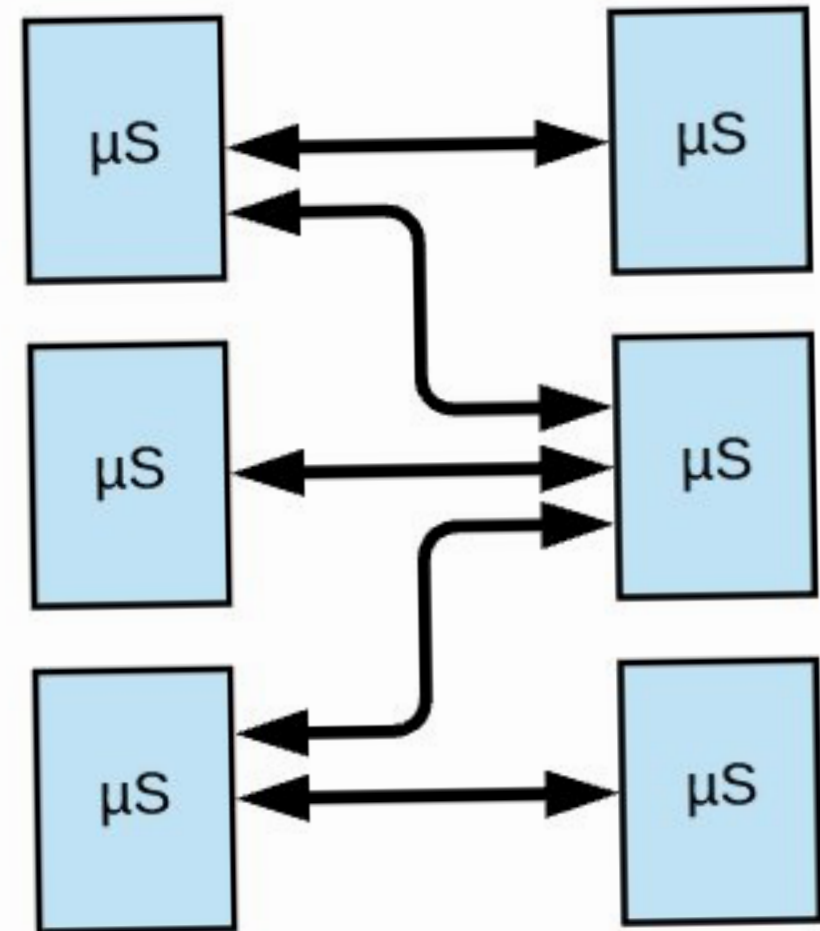
Daniel Bryant
@danielbryantuk

Following

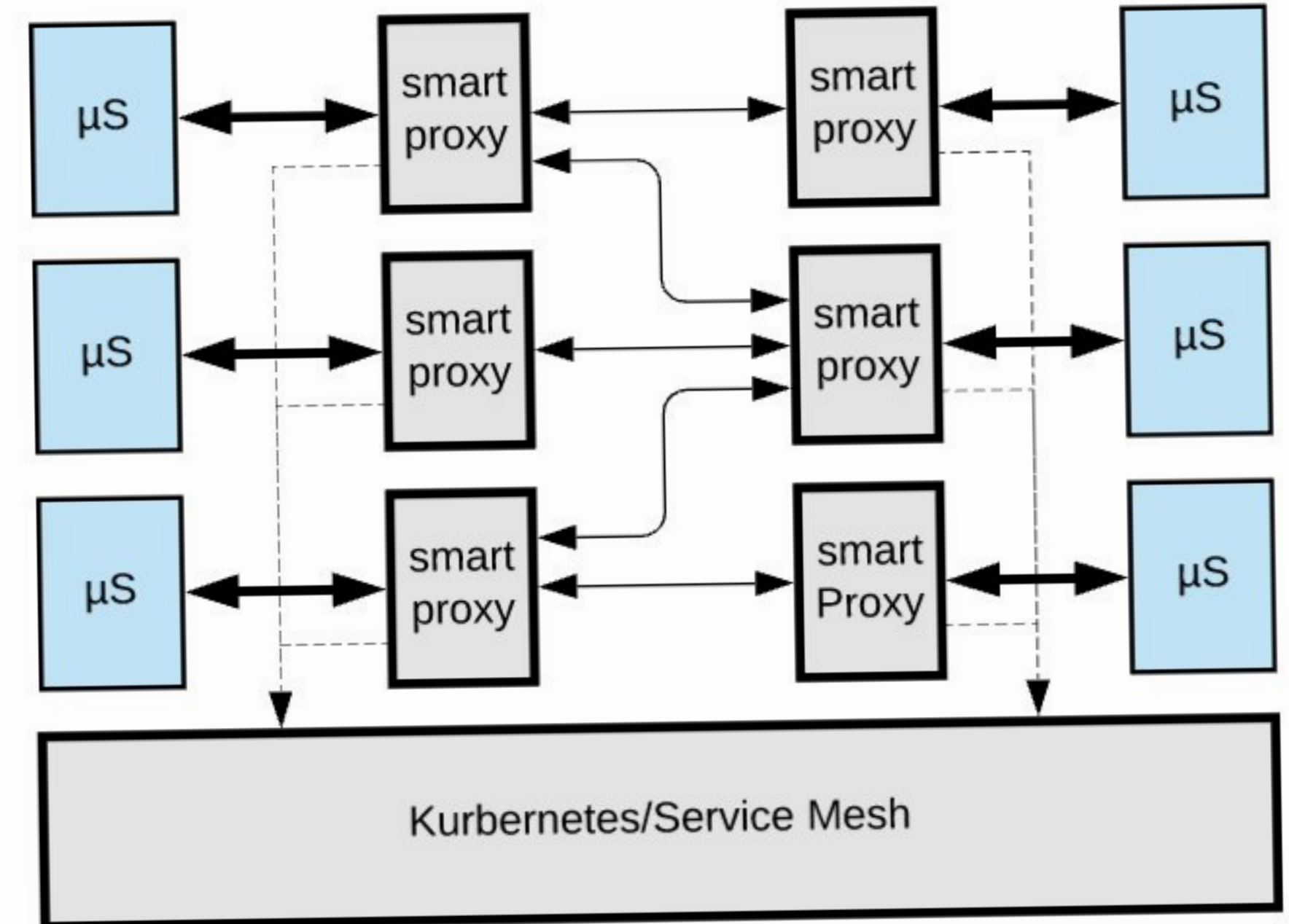
SOA/ESB
(smart pipes, dumb endpoints)



Microservices
(smart endpoints, dumb pipes)



Cloud Native
(smart platform, dumb services)



(a great article, btw)

#IBMCloudGarage

The screenshot shows a web browser window displaying an article on the InfoQ website. The browser's address bar shows the URL www.infoq.com/articles/microservices-post-kubern. The InfoQ logo is in the top left, with navigation links for Development, Architecture & Design, AI, ML & Data Engineering, Culture & Methods, DevOps, and Videos with Transcripts. A banner for the Software Development Conference in San Francisco (Nov 5-9, 2018) and London (Mar 4-8, 2019) is visible in the top right. Below the navigation is a breadcrumb trail: You are here: [InfoQ Homepage](#) > [Articles](#) > [Microservices in a Post-Kubernetes Era](#). The article title is "Microservices in a Post-Kubernetes Era" with a thumbnail image of a boat. Below the title, it says "Like" (with a thumbs up icon), "Posted by Bilgin Ibryam, reviewed by Daniel Bryant on Sep 01, 2018. Estimated reading time: 8 minutes / 1 Discuss". There are social sharing icons for Twitter, YouTube, Reddit, Facebook, and Email. On the right, there are buttons for "Reading List" and "Read later". The main content area has a section titled "Key Takeaways" with a bulleted list. The right sidebar has a "RELATED CONTENT" section with three article links and author photos.

InfoQ 1,201,360 Aug unique visitors


Development Architecture & Design AI, ML & Data Engineering Culture & Methods DevOps New Videos with Transcripts


Software Development Conference
San Francisco Nov 5-9, 2018
London Mar 4 - 8, 2019


Streaming Machine Learning Reactive Microservices Containers .NET *All topics* [The Architects' Newsletter](#)

You are here: [InfoQ Homepage](#) > [Articles](#) > [Microservices in a Post-Kubernetes Era](#)

Microservices in a Post-Kubernetes Era



 Like / Posted by [Bilgin Ibryam](#), reviewed by [Daniel Bryant](#) on Sep 01, 2018. Estimated reading time: 8 minutes / [1 Discuss](#)



Share 

[Reading List](#) [Read later](#)

Key Takeaways

- The microservice architecture is still the most popular architectural style for distributed systems. But Kubernetes and the cloud native movement has redefined certain aspects of application design and development at scale.
- On a cloud native platform, observability of services is not enough. A more fundamental prerequisite is to make microservices automatable, by implementing health checks, reacting to signals, declaring resource consumption, etc.
- In the post-Kubernetes era, using libraries to implement operational networking

RELATED CONTENT

- [eBay Replatforming to Kubernetes, Envoy and Kafka: Intending to Open Source Hardware and Software](#) Sep 16, 2018
- [Networking Your Microservices Applications](#) Sep 21, 2018 
- [Designing Events-First Microservices](#) Sep 13, 2018 

www.cncf.io

CLOUD NATIVE
COMPUTING FOUNDATION

About Projects Certification People Community Newsroom

JOIN NOW

contributors to CNCf projects

for free Kubernetes EdX course

Kubernetes Distributions and Platforms

Meetup members

What is CNCf?

CNCf is an open source software foundation dedicated to making cloud native computing universal and sustainable. Cloud native computing uses an open source software stack to deploy applications as microservices, packaging each part into its own container, and dynamically orchestrating those containers to optimize resource utilization. Cloud native technologies enable software developers to build great products faster.

JOIN

www.cncf.io

CLOUD NATIVE COMPUTING FOUNDATION

About Projects Certification People Community Newsroom

JOIN NOW

contributors to CNCf projects

for free Kubernetes EdX course

Kubernetes Distributions and Platforms

Meetup members

What is CNCf?

microservices containers dynamically orchestrated

JOIN

Twitter, Facebook, YouTube, RSS, Chat

DevoXX

"the cloud native
computing
foundation is wrong
...
about cloud native."



DevoXX

"the cloud native
computing
foundation is wrong
...
about cloud native."



Holly

www.cncf.io

CLOUD NATIVE
COMPUTING FOUNDATION

About Projects Certification People Community Newsroom

JOIN NOW

contributors to CNCF projects

for free Kubernetes EdX course

Kubernetes Distributions and Platforms

Meetup members

What is CNCF?

CNCF is an open source software foundation dedicated to making cloud native computing universal and sustainable. Cloud native computing uses an open source software stack to deploy applications as microservices, packaging each part into its own container, and dynamically orchestrating those containers to optimize resource utilization. Cloud native technologies enable software developers to build great products faster.

JOIN

www.cncf.io

CLOUD NATIVE COMPUTING FOUNDATION

About Projects Certification People Community Newsroom

JOIN NOW

contributors to CNCf projects

for free Kubernetes EdX course

Kubernetes Distributions and Platforms

Meetup members

What is CNCf?

build great products faster

JOIN

Twitter Facebook YouTube RSS Share

why?

what **problem** are
we trying to solve?



CV-driven development.



“my CV looks dull”
is not a good reason
to go cloud native

why cloud?

cost





e l a s t i c i t y

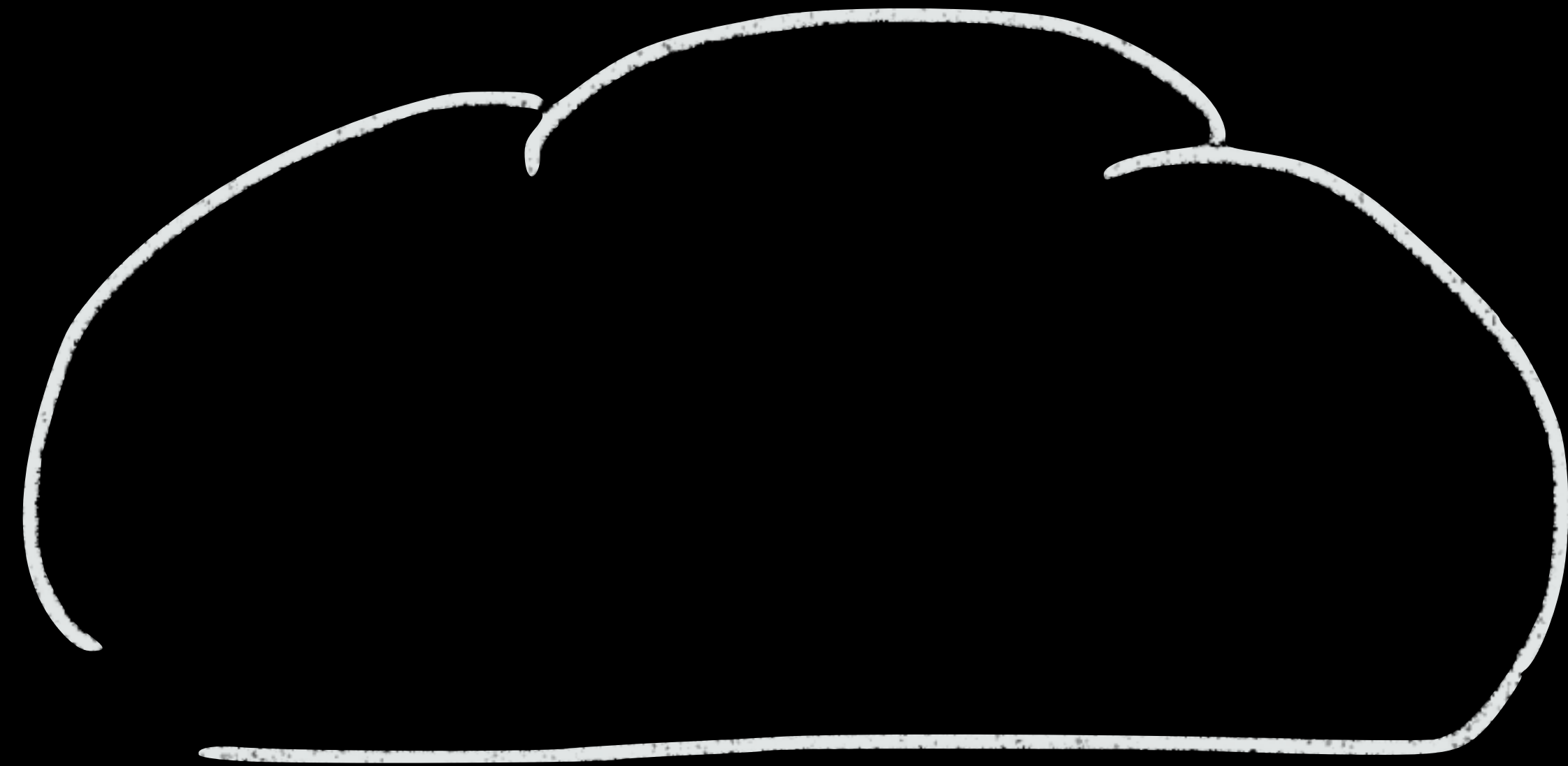
Speed



exotic capabilities



why cloud native?



12

factors

12 factors

how to write a
cloud application
so you don't get
electrocuted

#IBMCloudGarage

@holly_cummins

#IBMCloudGarage

@holly_cummins

cloud native is not a
synonym for
'microservices'

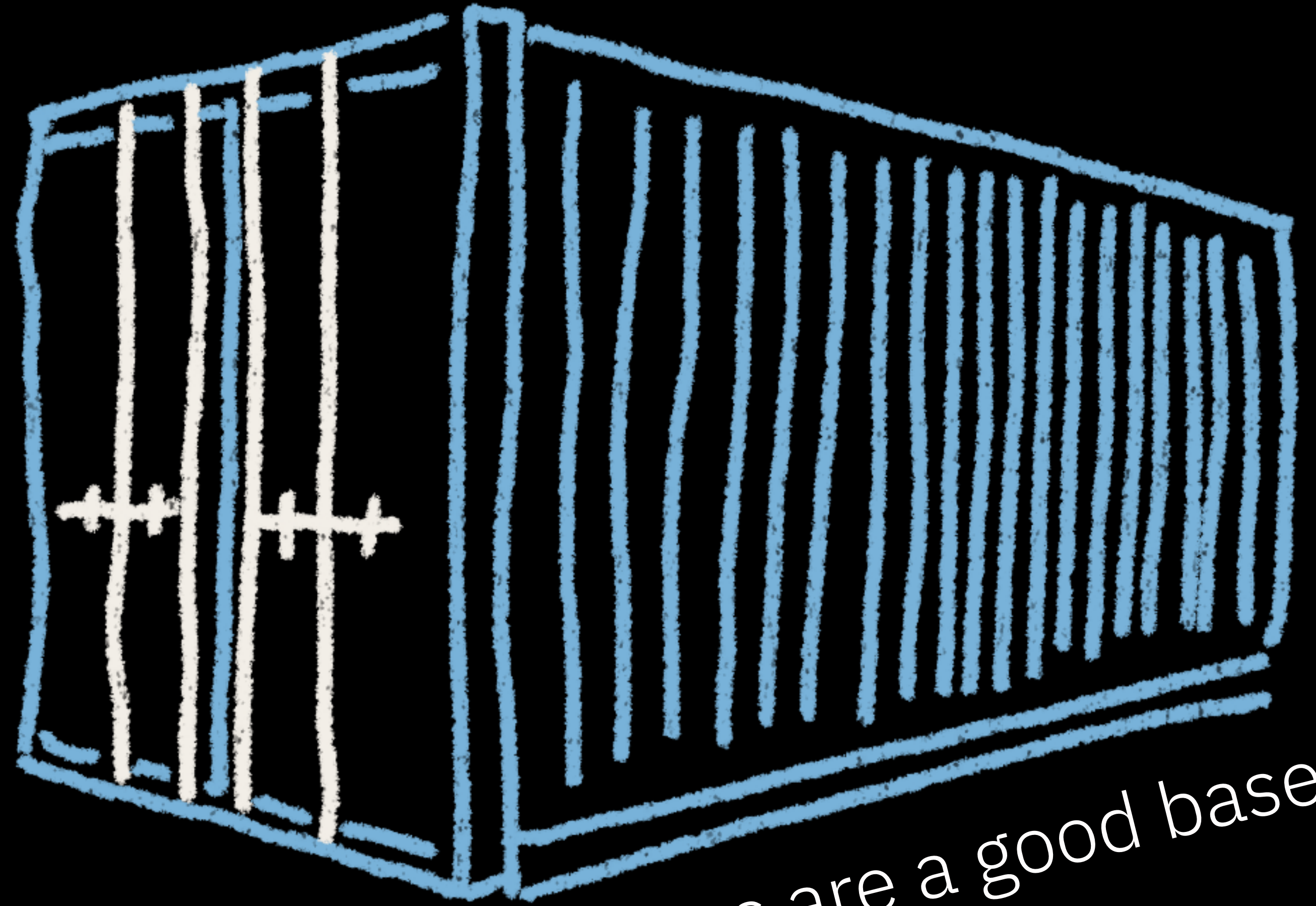
#IBMCloudGarage

@holly_cummins

if 'cloud native' has to be a
synonym for anything, it would be
'idempotent'

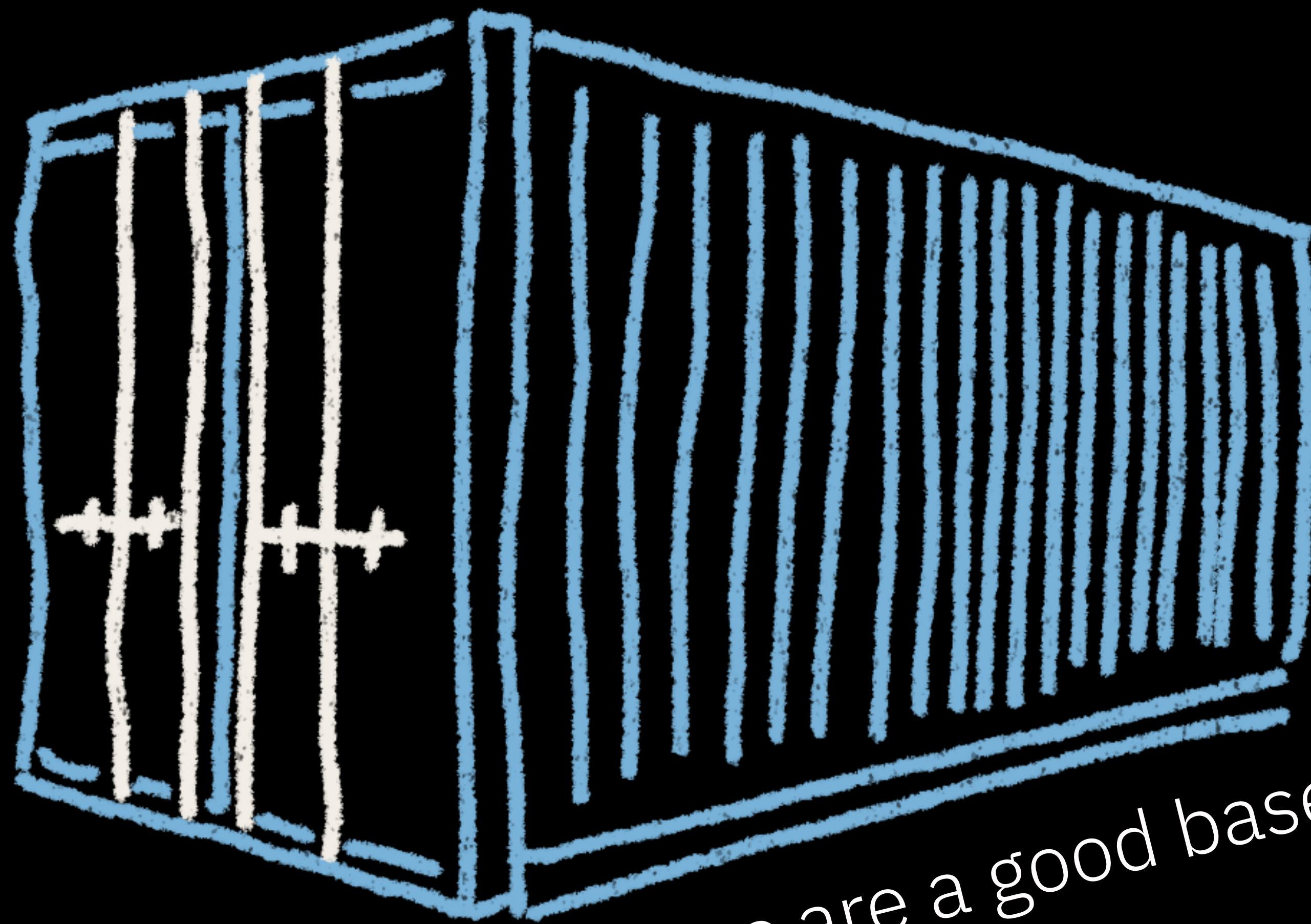
if 'cloud native' has to be a
synonym for anything, it would be
'idempotent'

which definitely needs a synonym



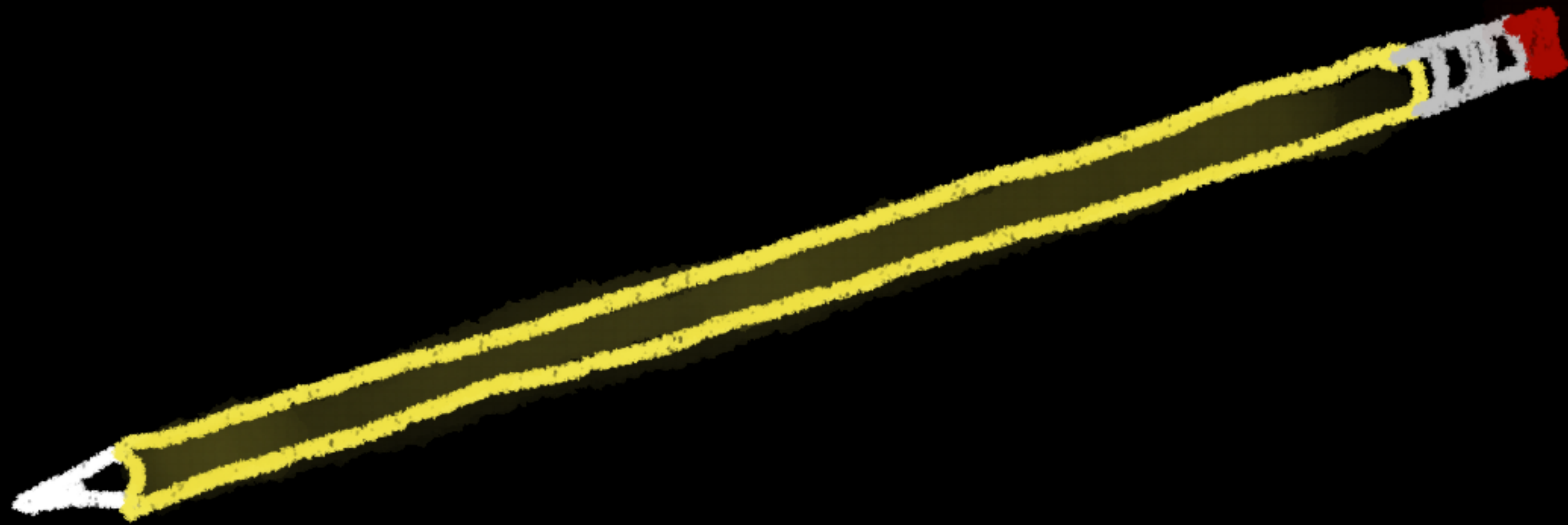
containers are a good base

it's not a
competition
to see how
many you
can have



containers are a good base

you do not need intra-app http
communication to be cloud native



complexity adds expense

unnecessary complexity
adds unnecessary expense

#IBMCloudGarage

@holly_cummins

space pencil

space pencil

\$128.89

space pencil space pen

\$128.89

space pencil

\$128.89

space pen

\$2.39

space pencil

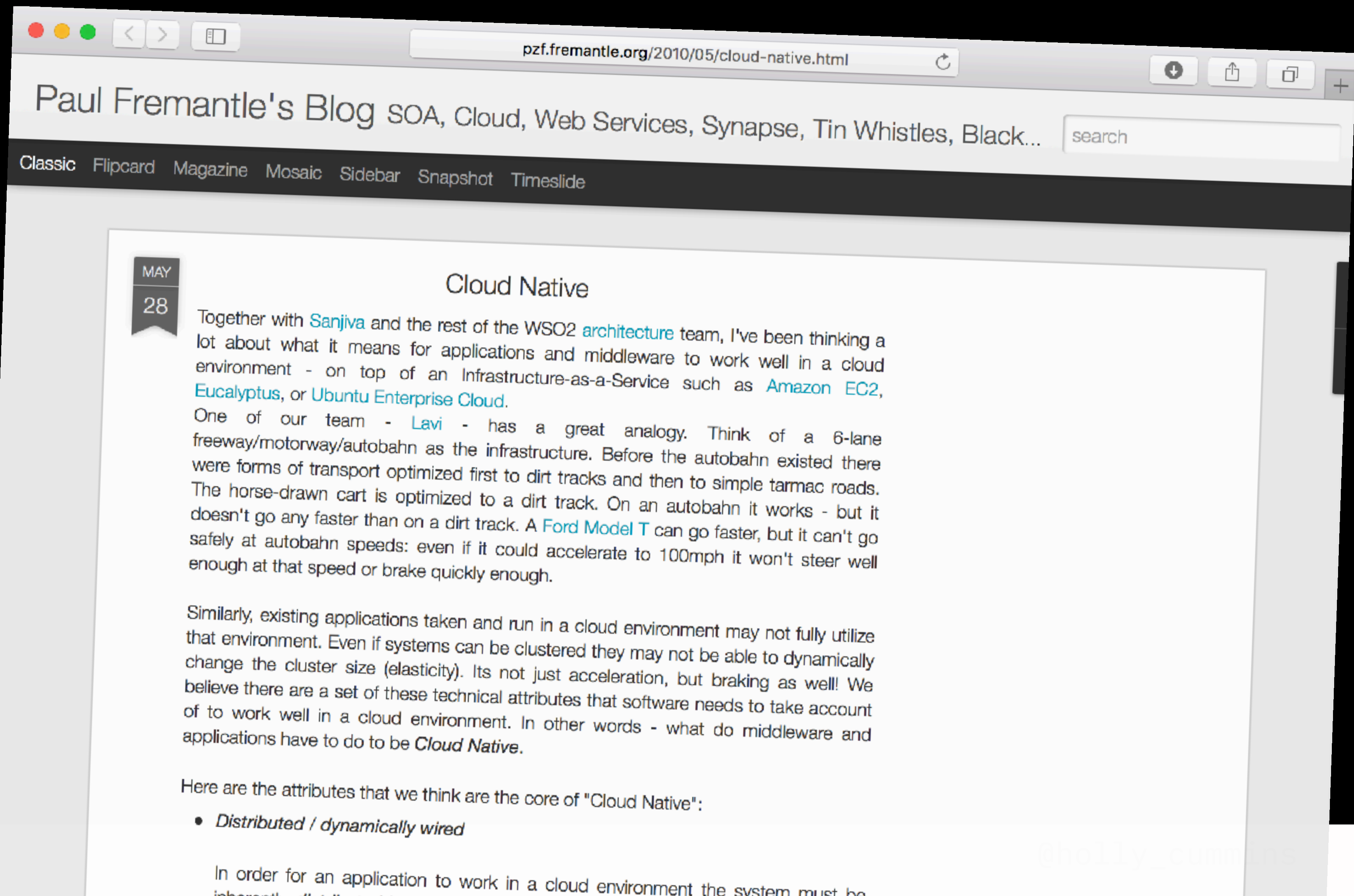
\$128.89

+medical bills

space pen

\$2.39

2010



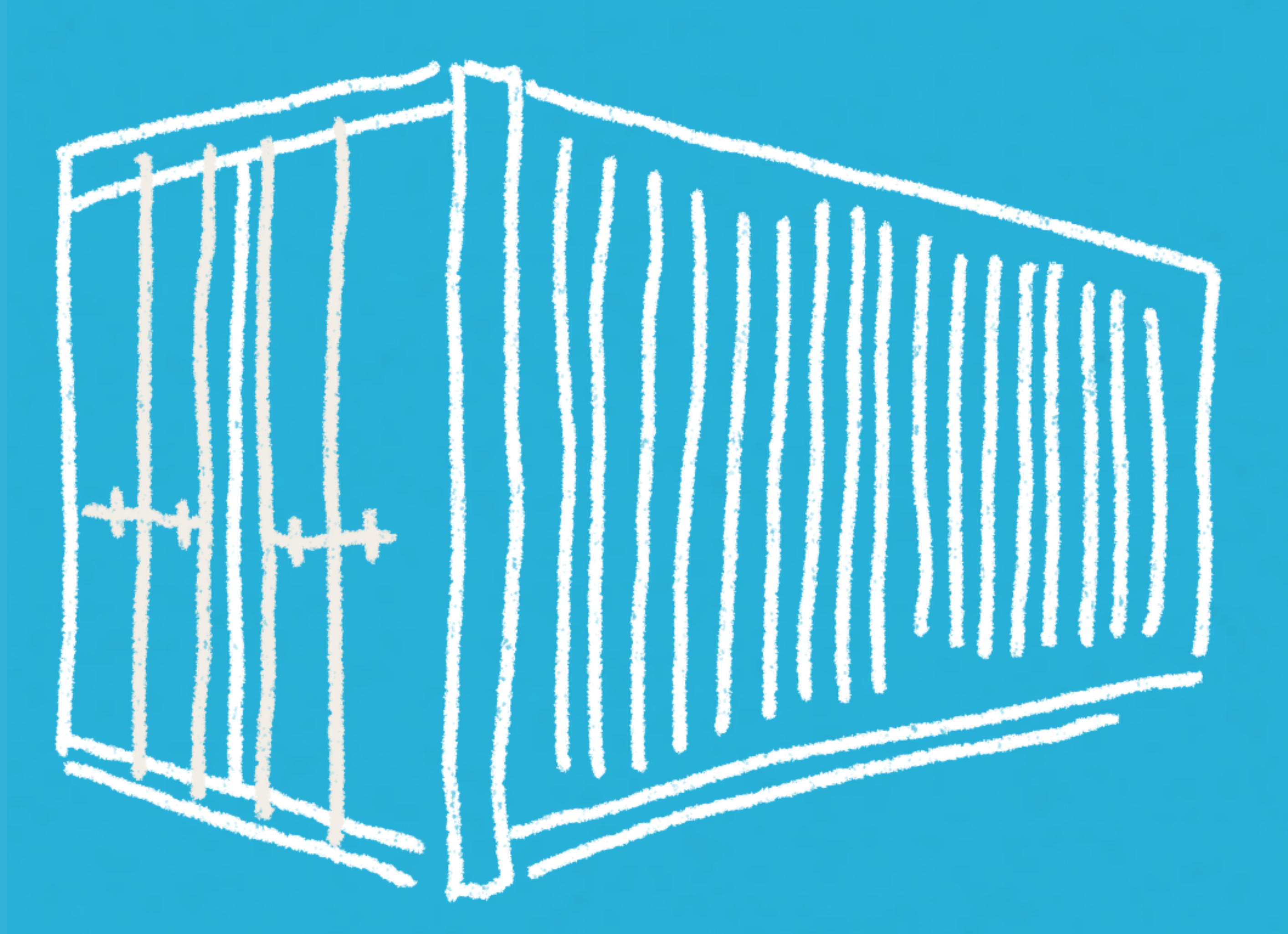
#IBMCloudGarage

#IBMCloudGarage

@holly_cummins

behaves well on the cloud

behaves well on the cloud
written for the cloud



what **is** a container?

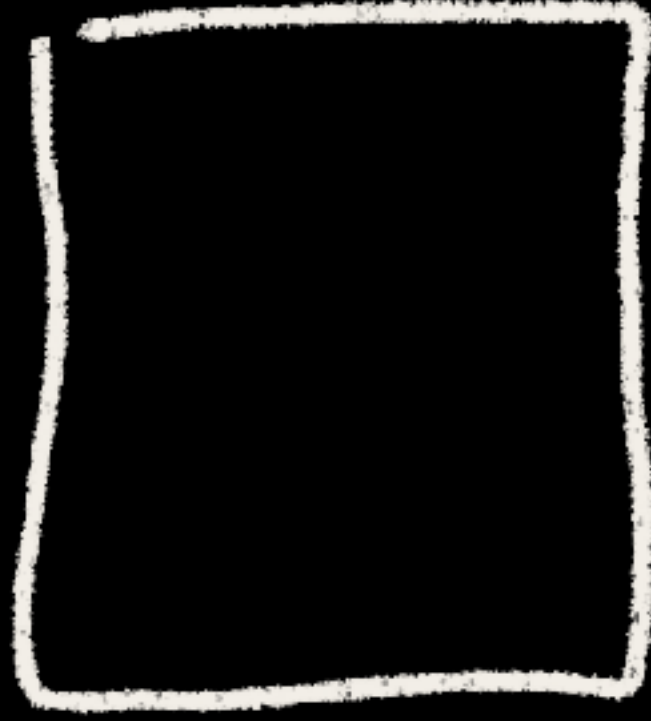
#IBMCloudGarage

@holly_cummins

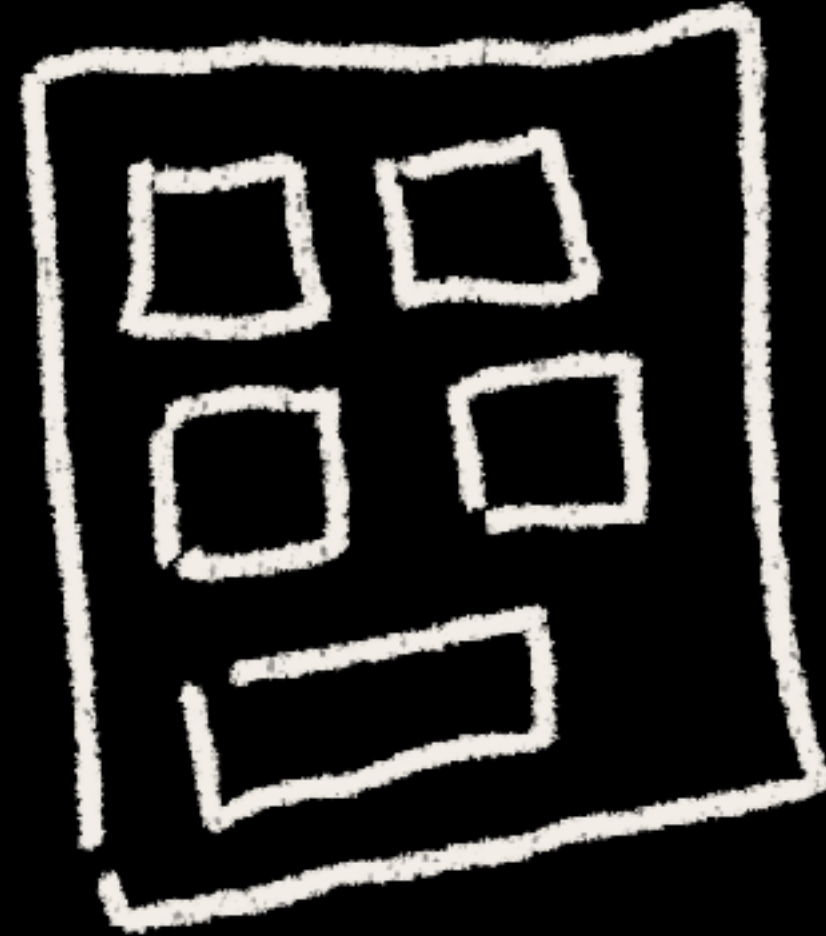
bare metal



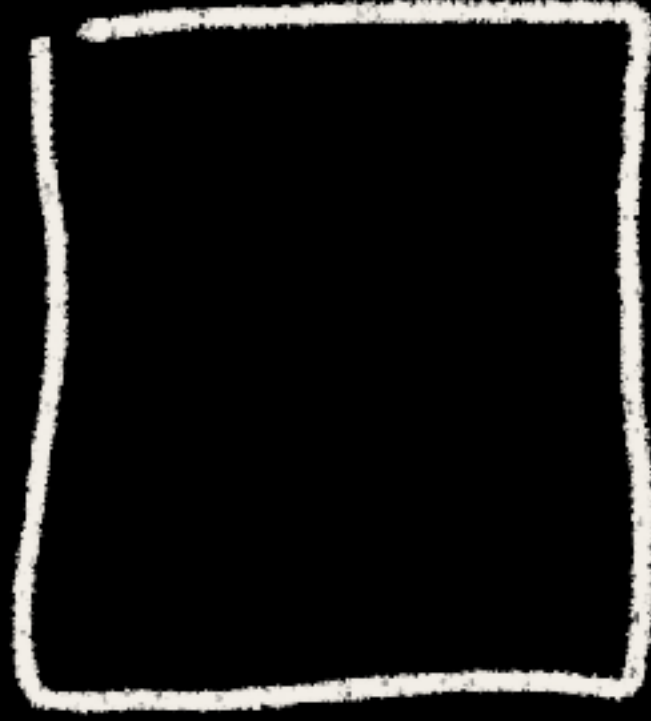
bare metal



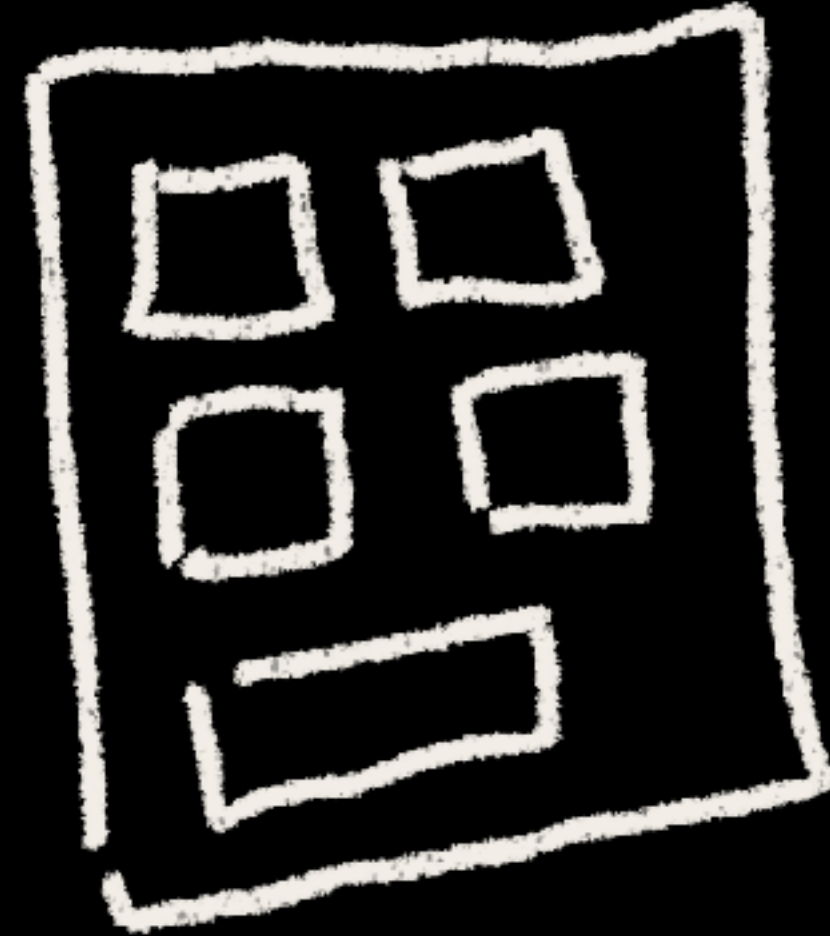
virtual machines



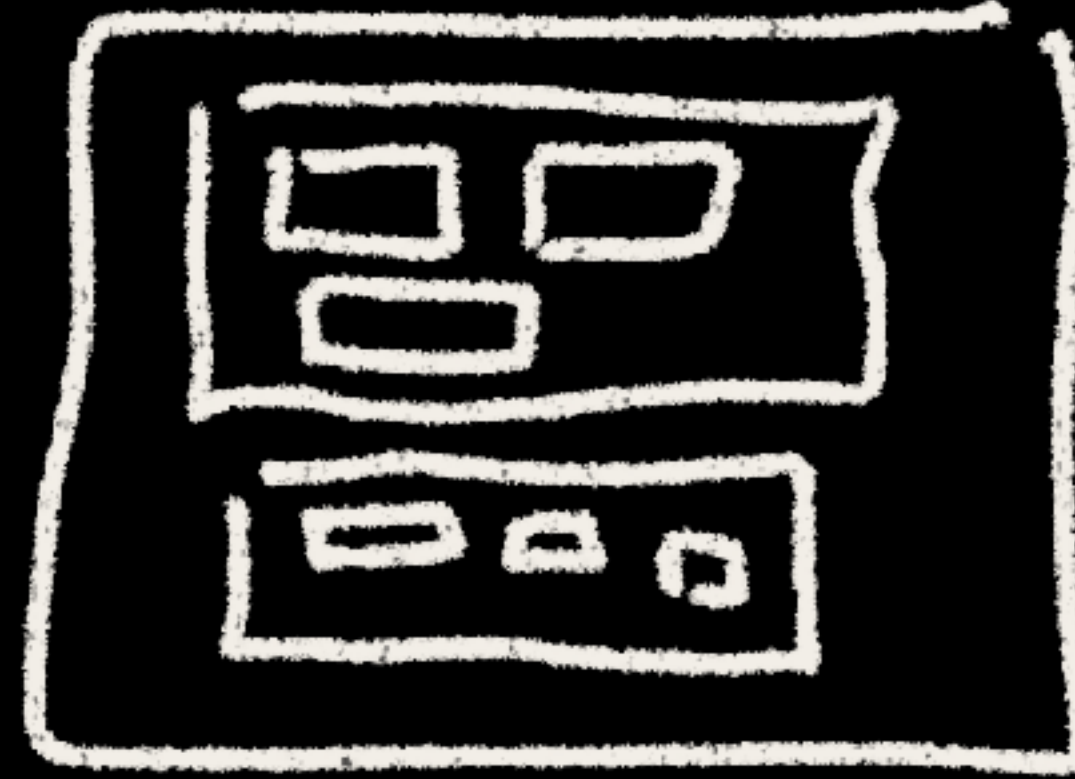
bare metal



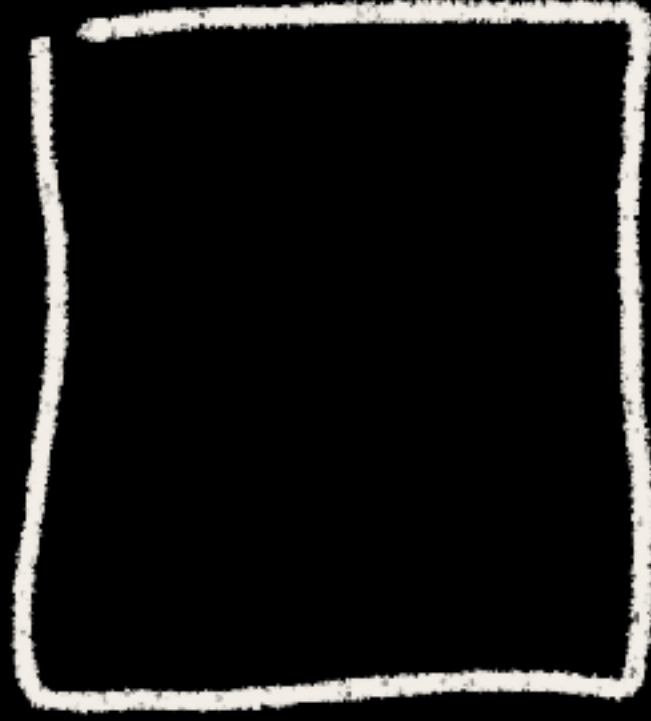
virtual machines



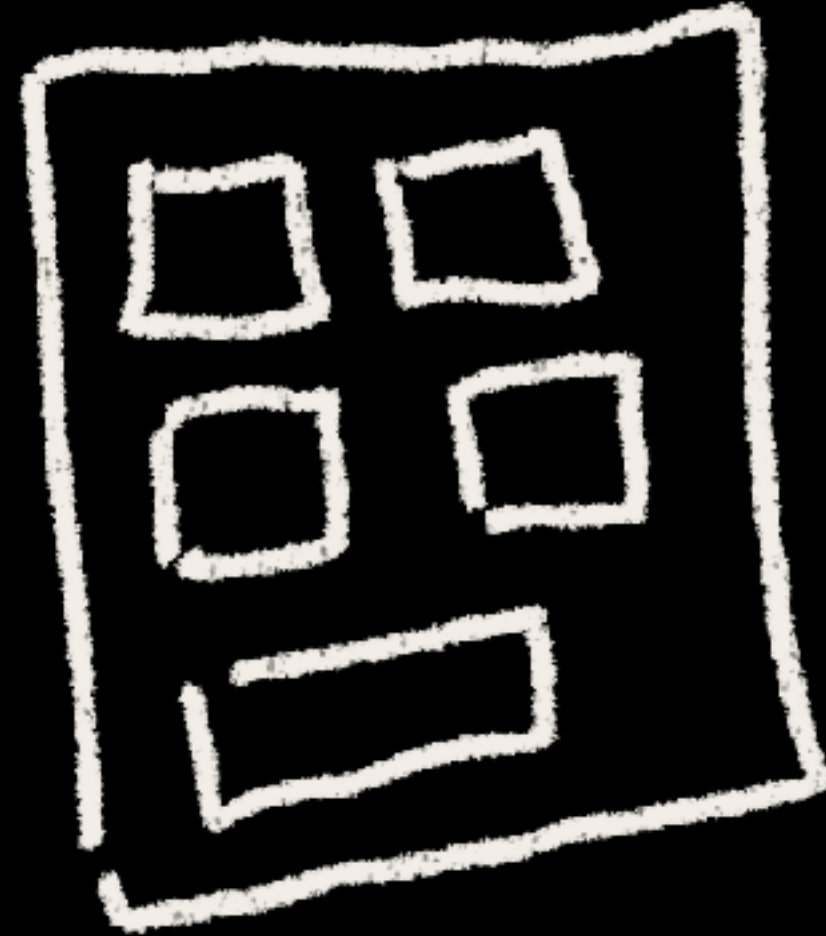
containers



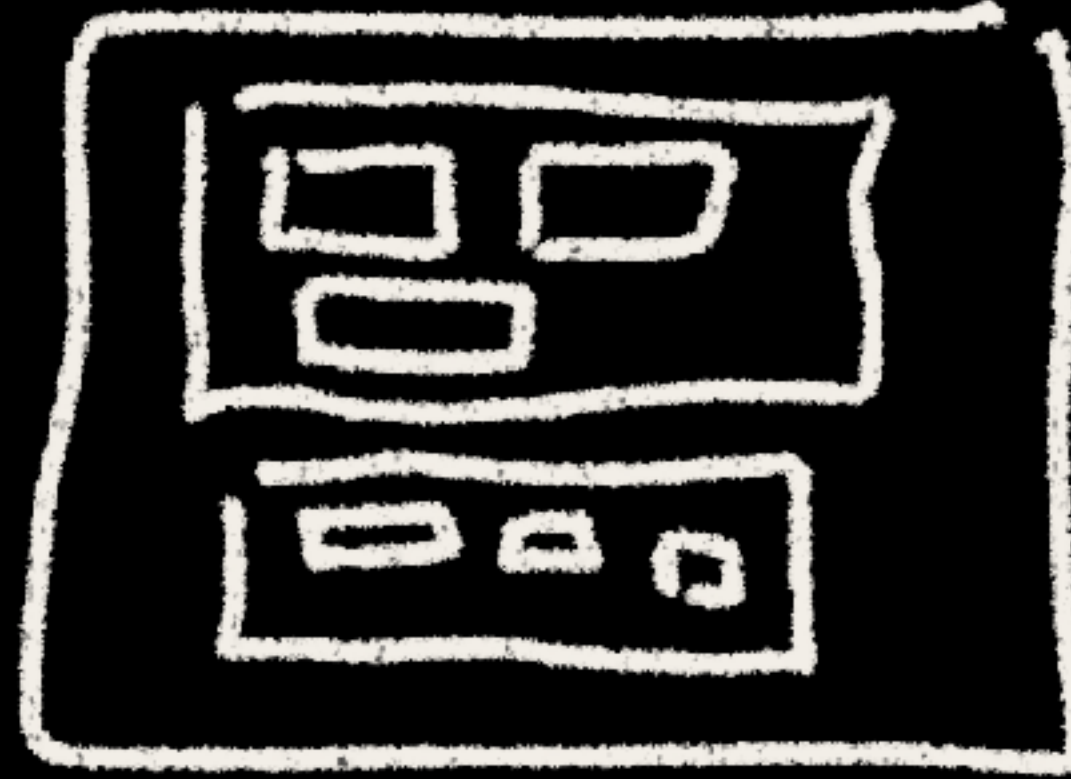
bare metal



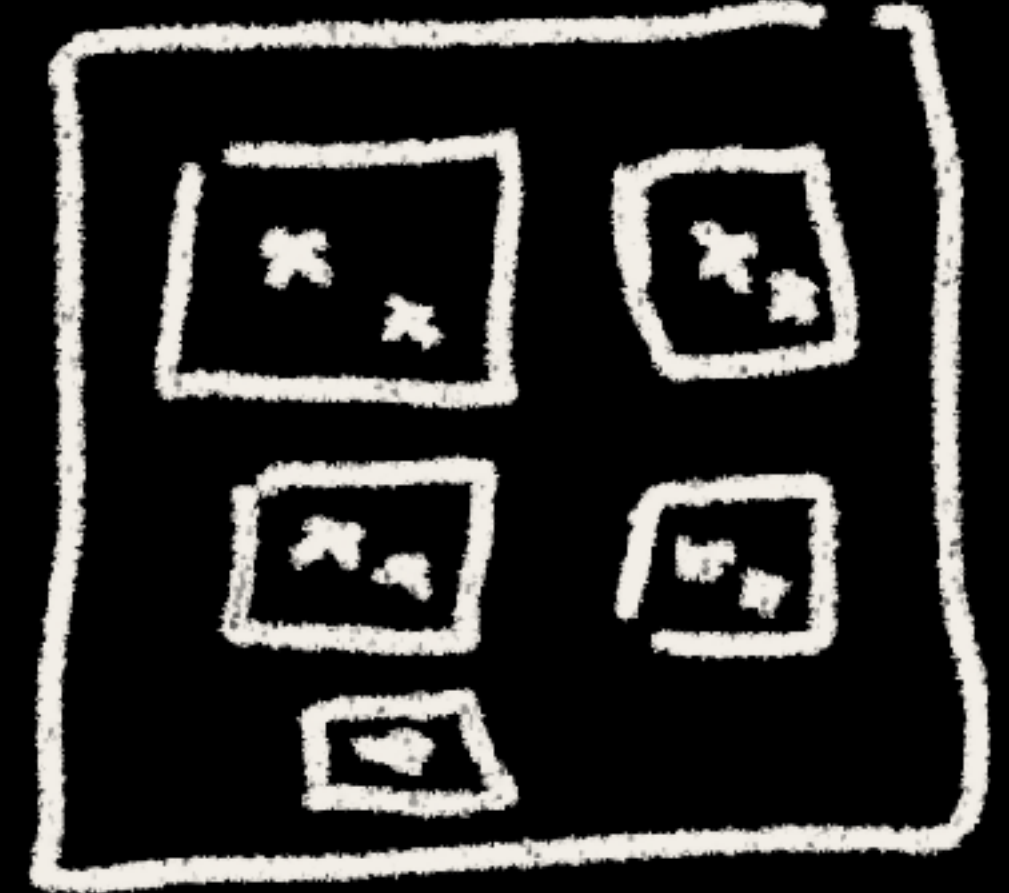
virtual machines



containers



functions



funtainers

function-tainers

funtainers



Thermos® FUNtainer®

funtainers

all the fun of containers, but without having to worry about kubernetes

this is all how we **run** our
application, not what's **in** it

speed

speed

what's the point of getting the
same old **stuff** to market faster?

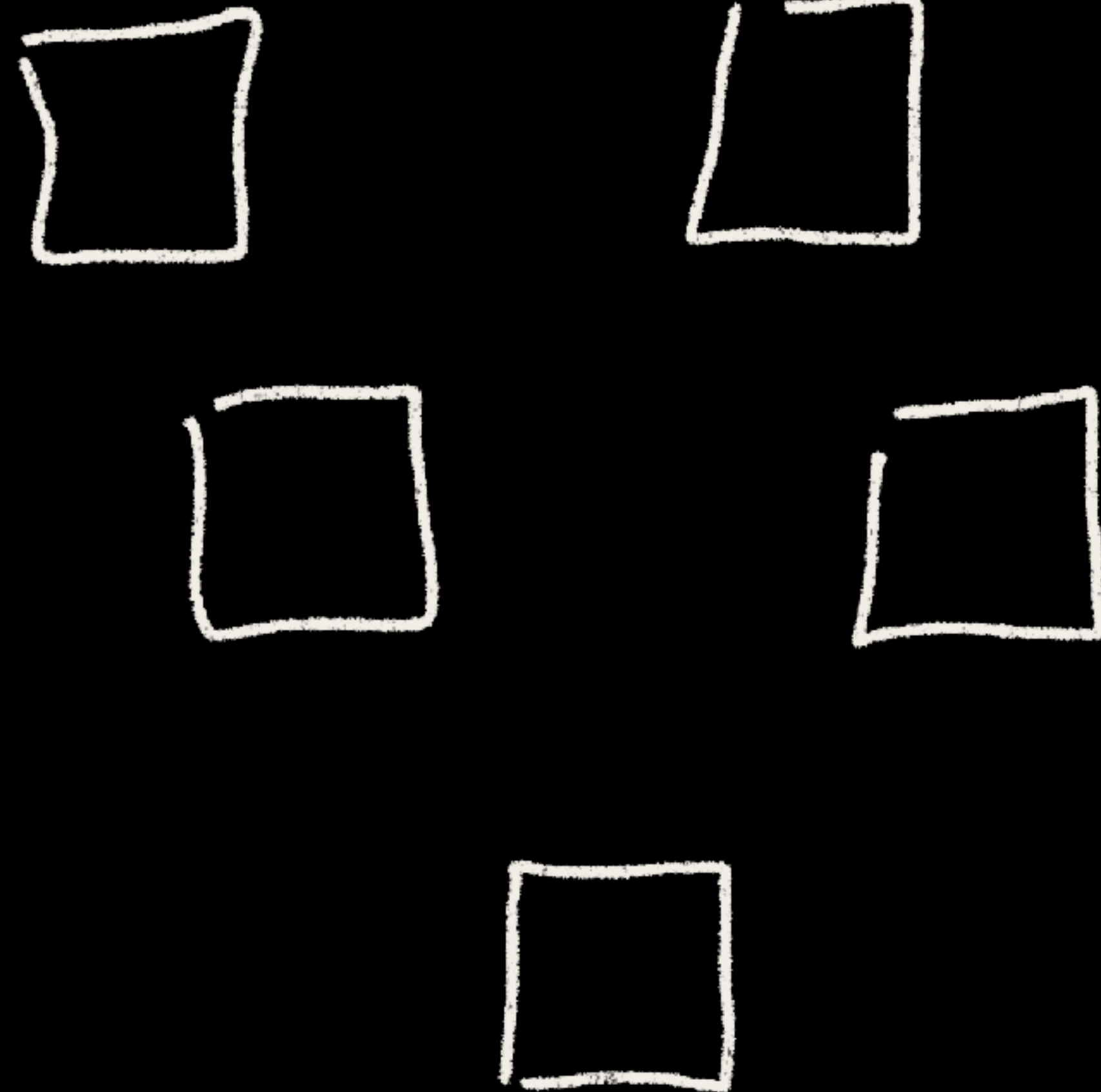
what's the point of being able to
respond to the market, if you **don't**?

what's the point of
architecture that can go
faster, if you don't go faster?

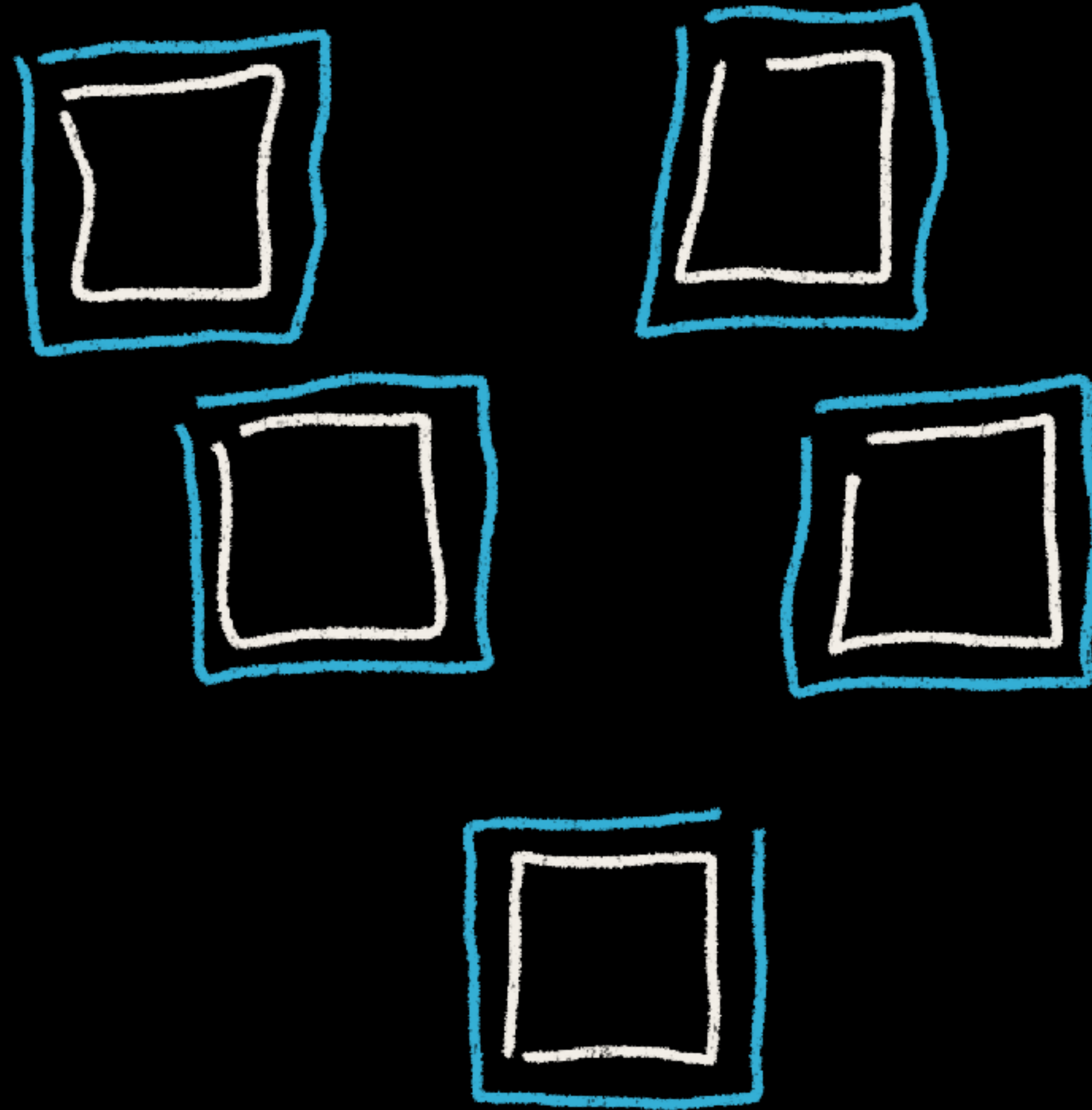
how to **fail** at cloud native



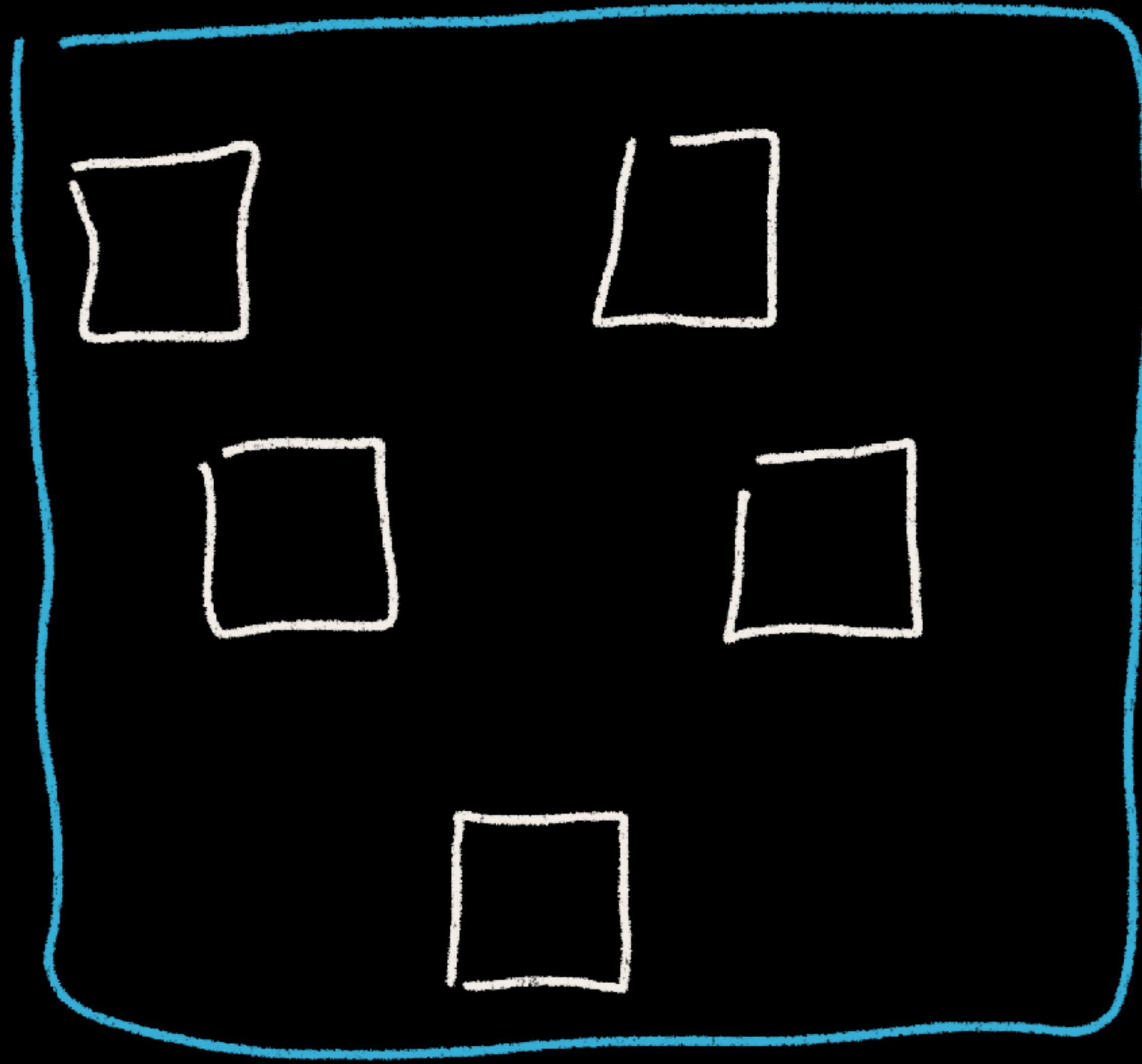
“each of our microservices has duplicated the same object model ... with twenty classes and seventy fields”



Microservice
Domain



Microservice
Domain



“every time we change
code, something breaks”

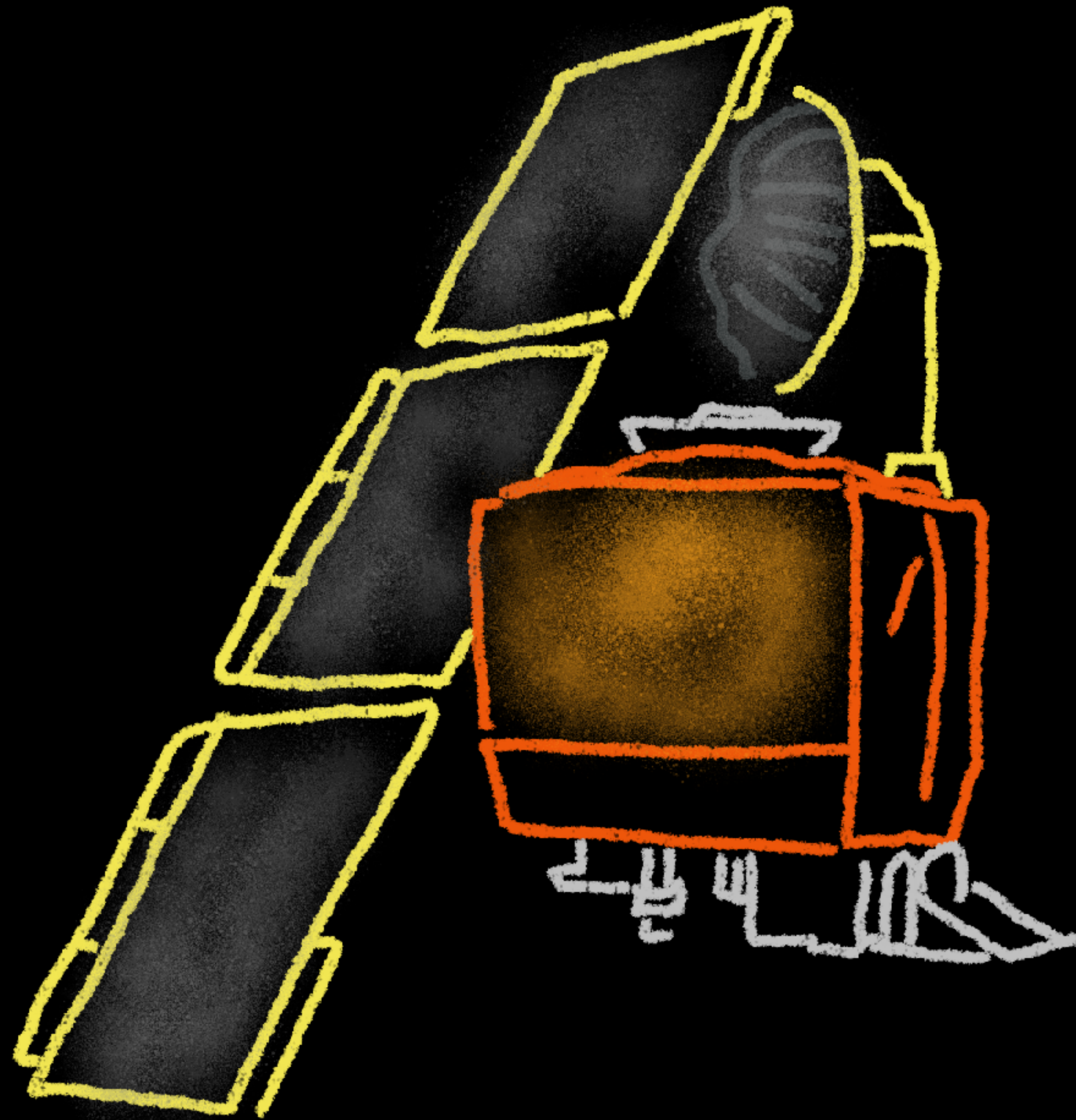
distributed monolith

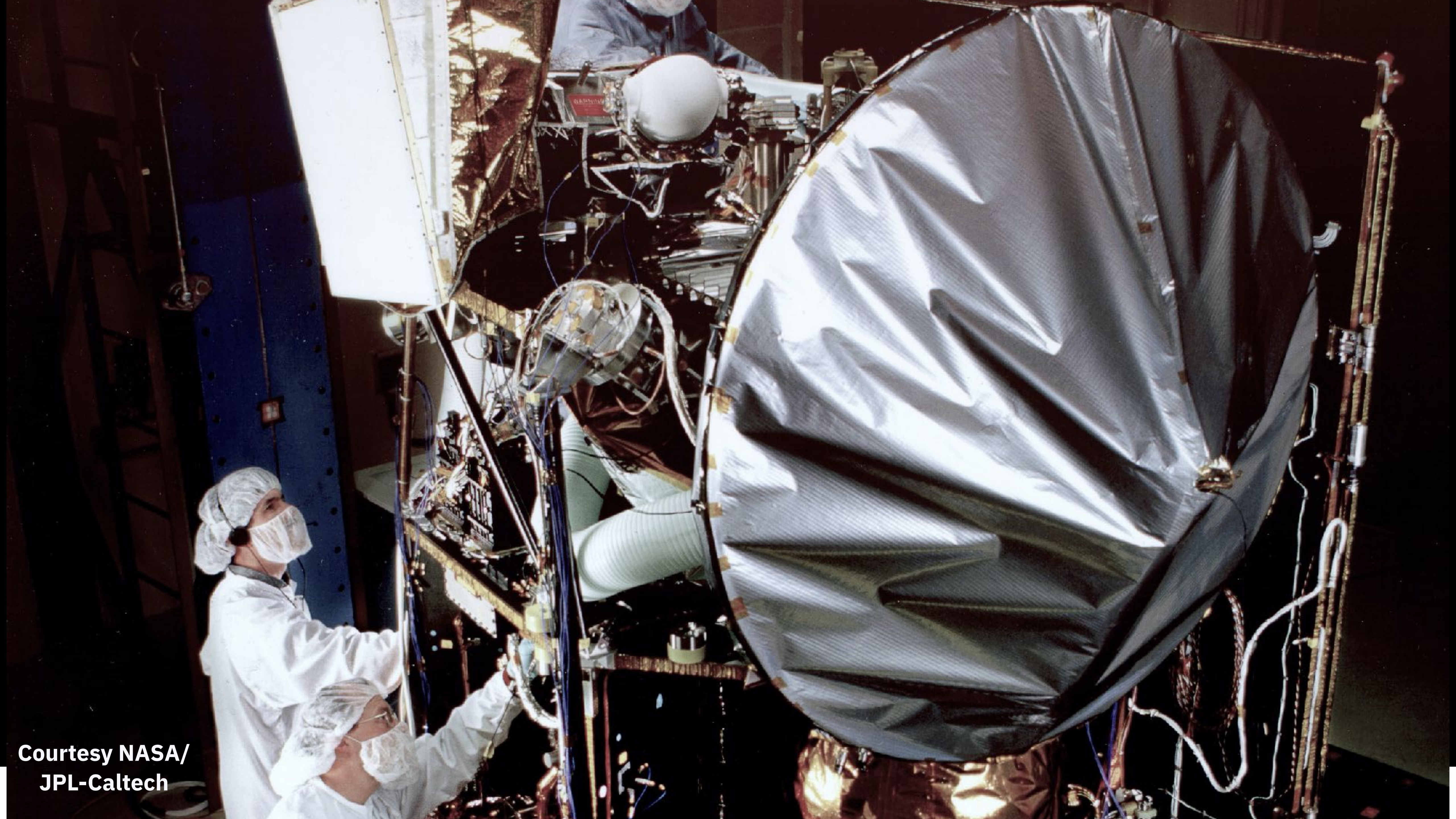


cloud-native spaghetti is still spaghetti

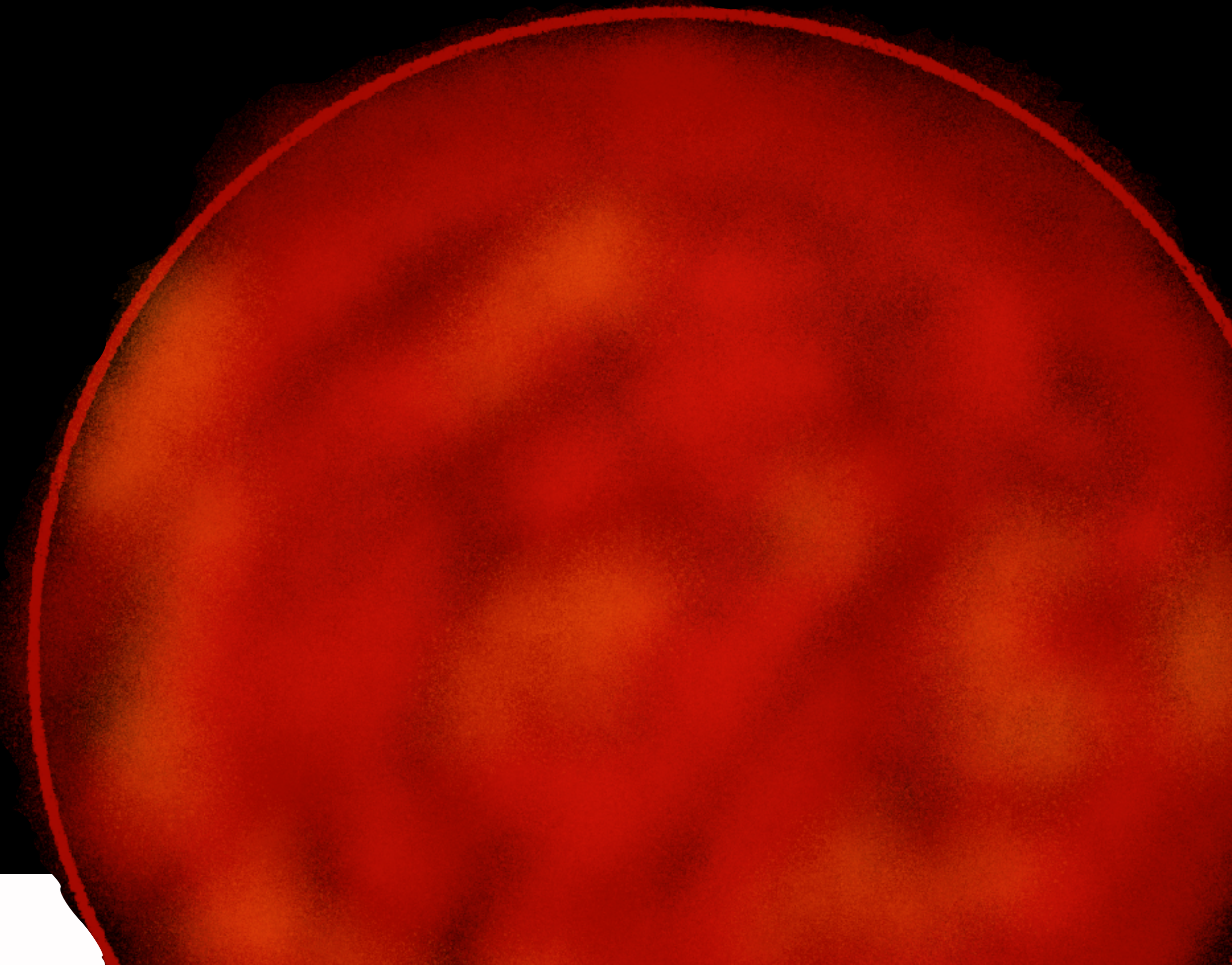
(Image: Cloudy with a Chance of Meatballs.)

just because a system runs across 6
containers doesn't mean it's decoupled

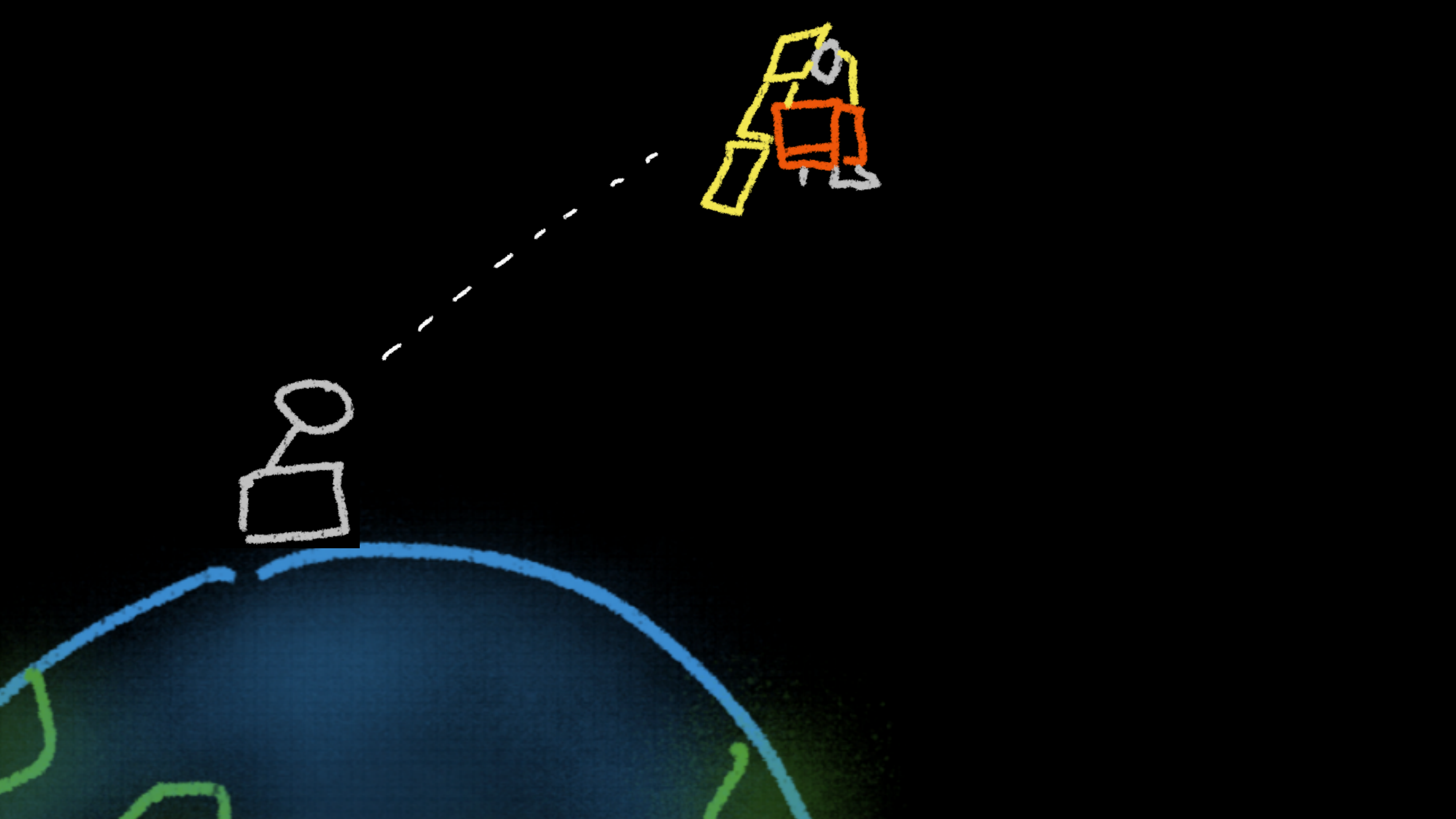


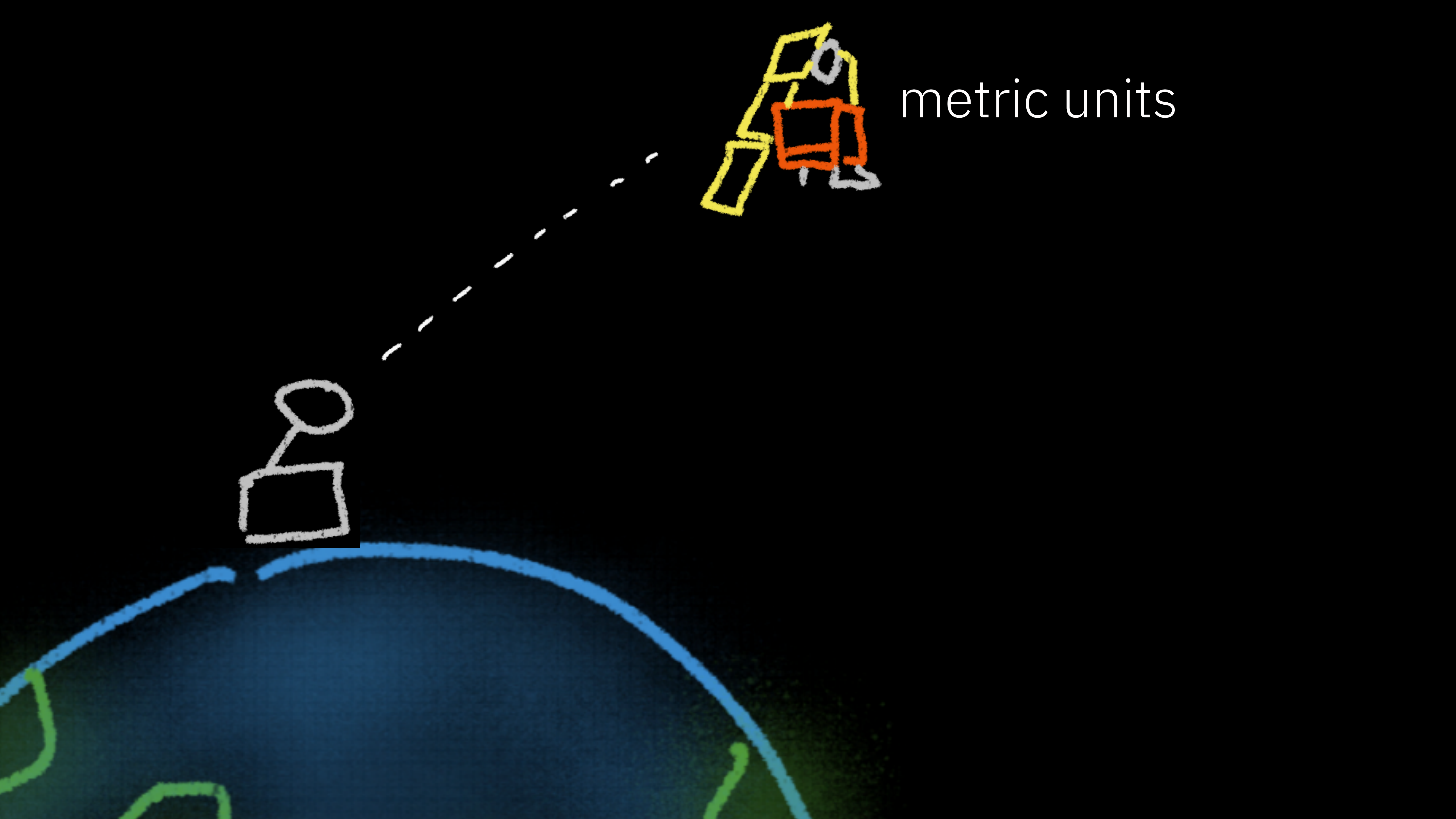


Courtesy NASA/
JPL-Caltech

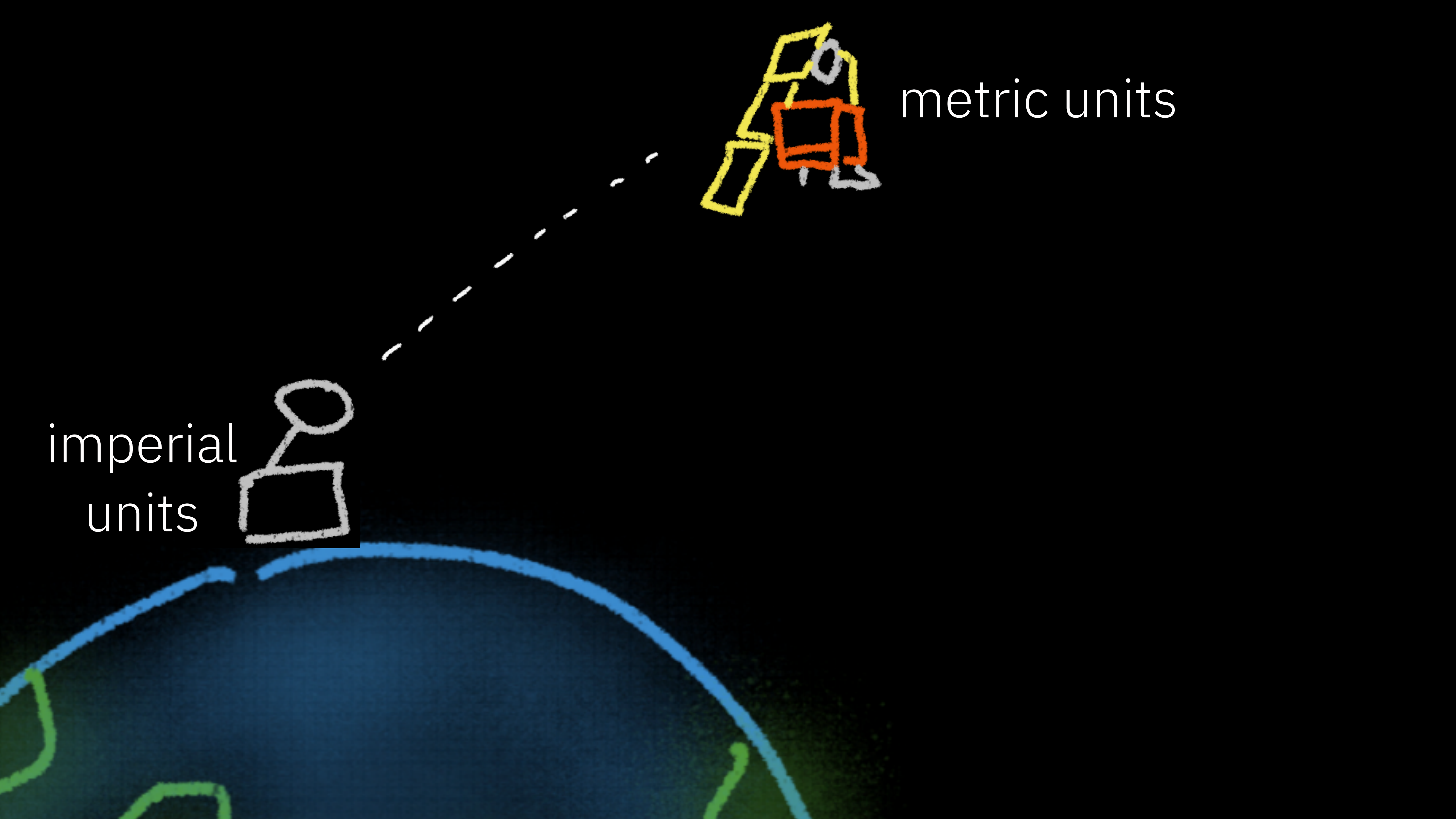


#IBMCloudGarage



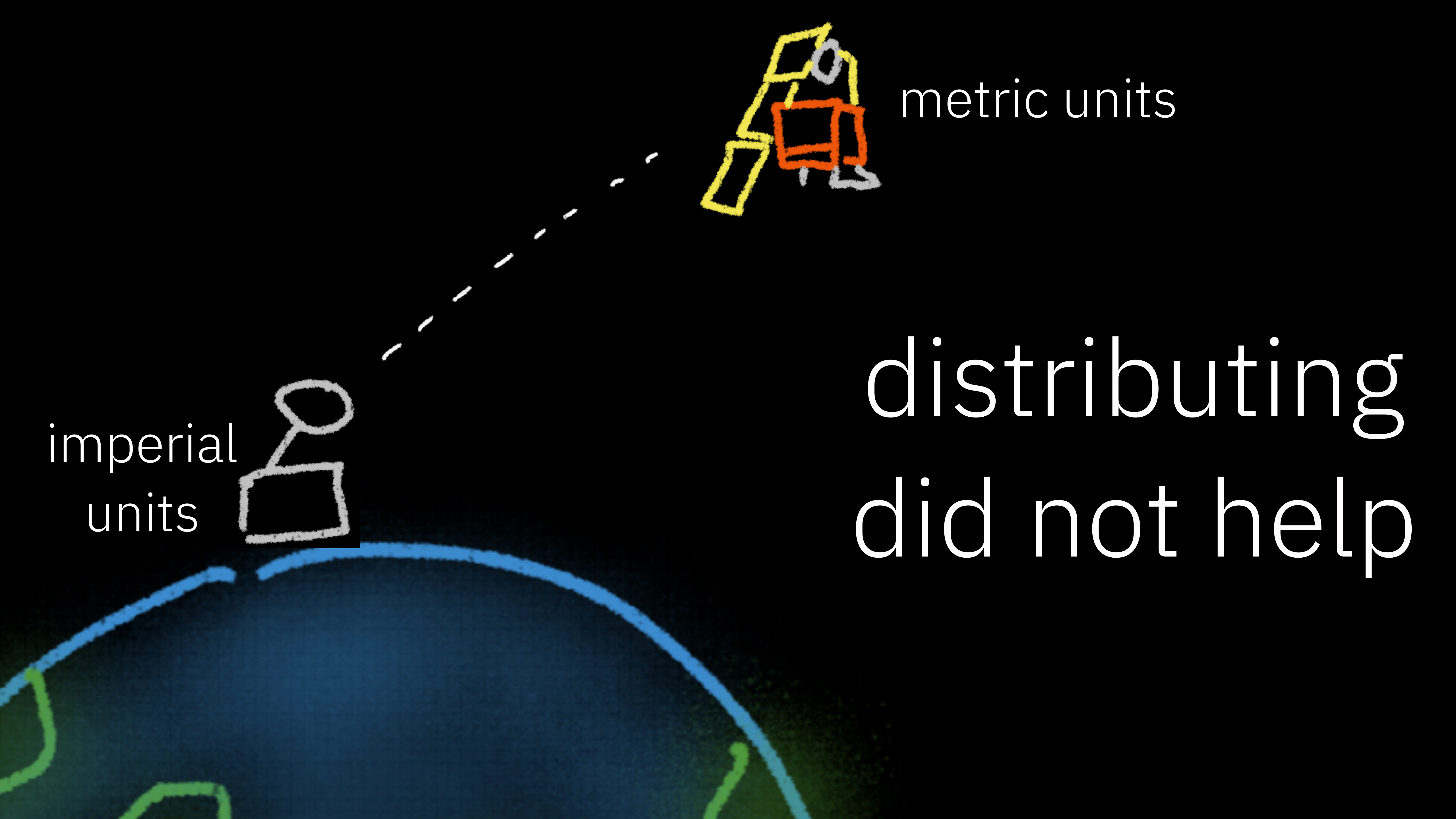


metric units



imperial
units

metric units



imperial
units



metric units

distributing
did not help

microservices **need**
consumer-driven contract tests

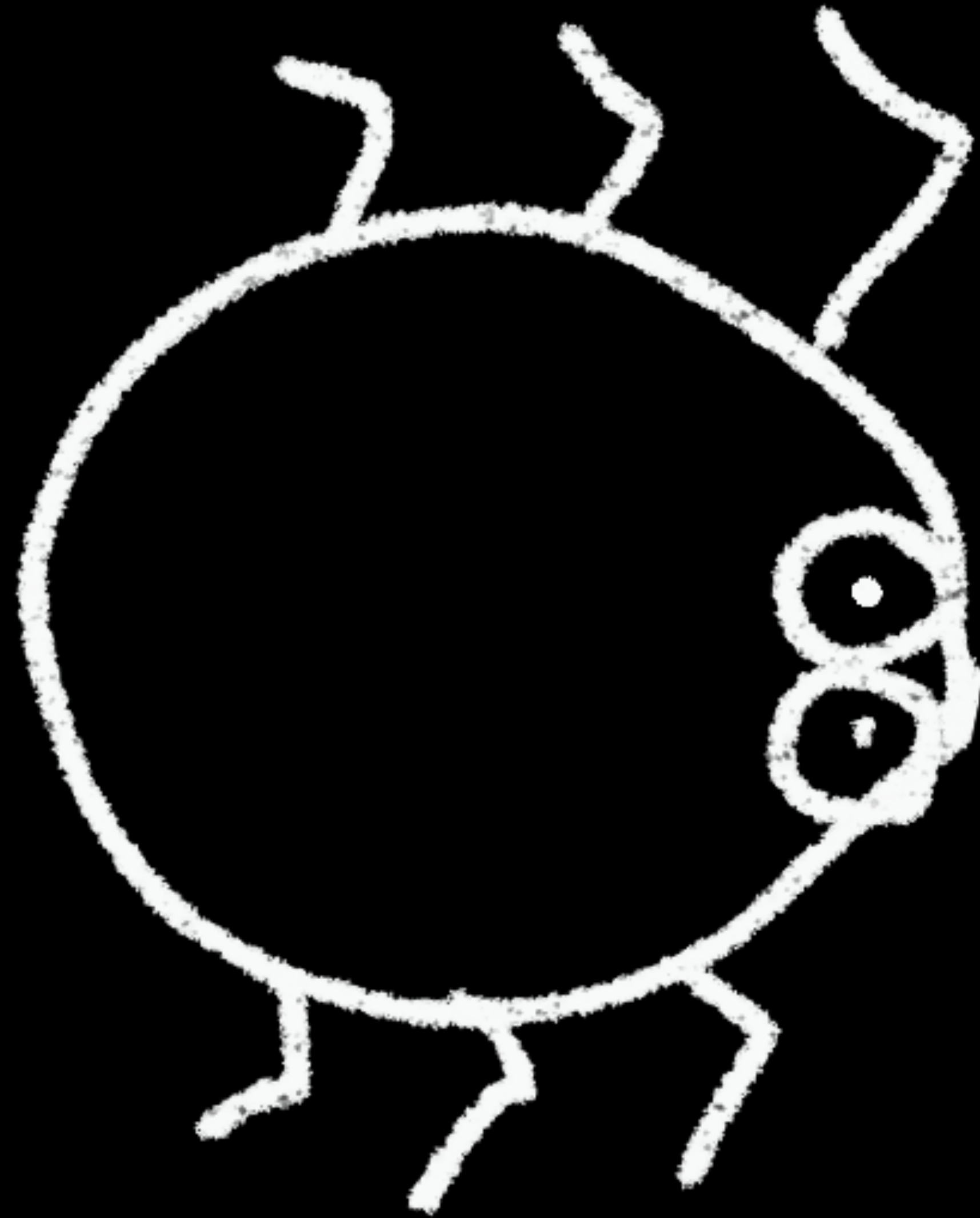
“our tests aren’t
automated”

“we don’t know if
our code works”

systems **will** behave in
unexpected ways

documentation can be
wrong

dependency updates
can change behaviour

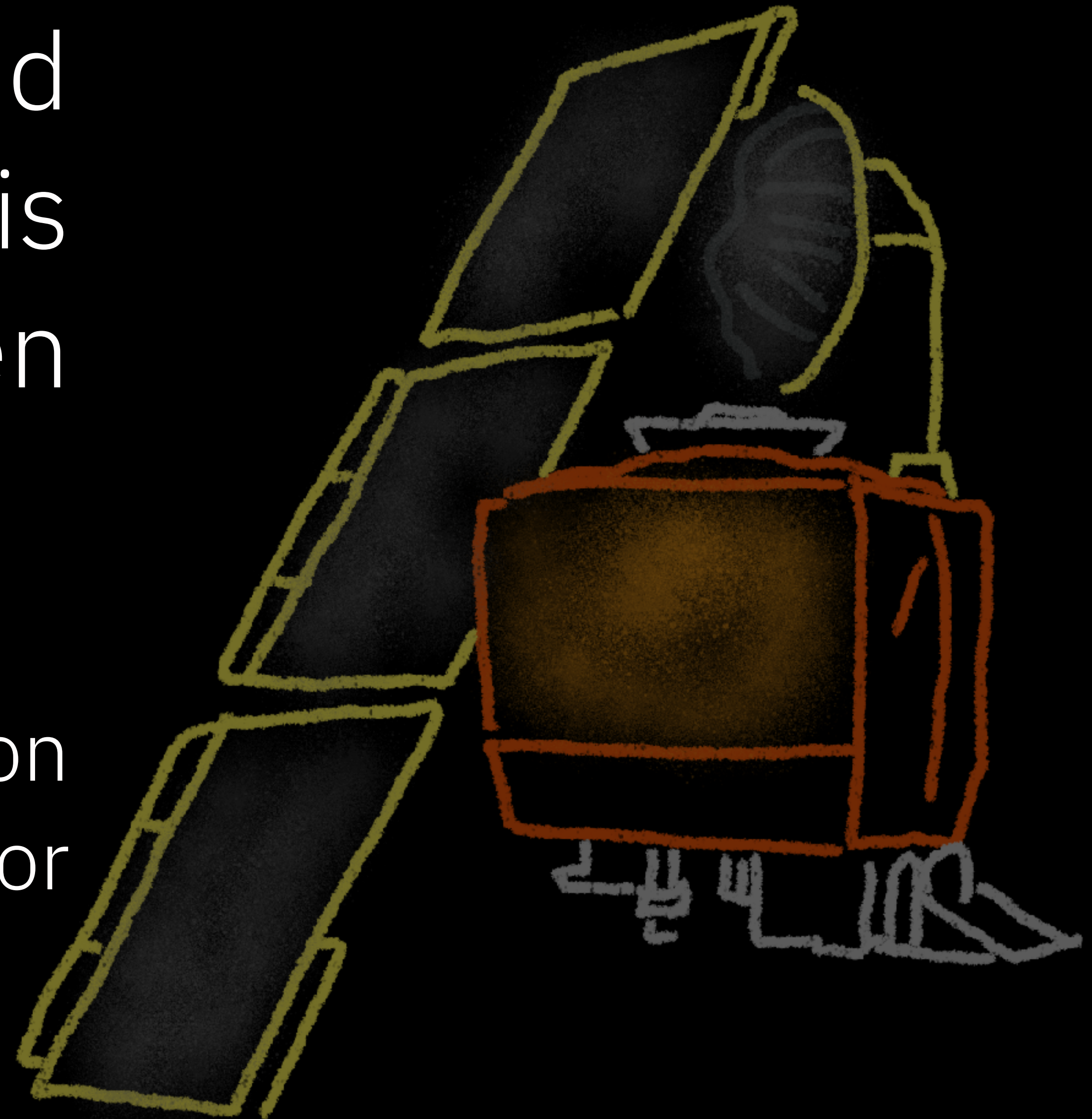


#IBMCloudGarage

@holly_cummins

“Had we done end-to-end testing, we believe this error would have been caught.”

Arthur Stephenson
Chief Investigator



“we can’t ship
until we have
more confidence
in the quality”



microservices **need**
automated integration tests

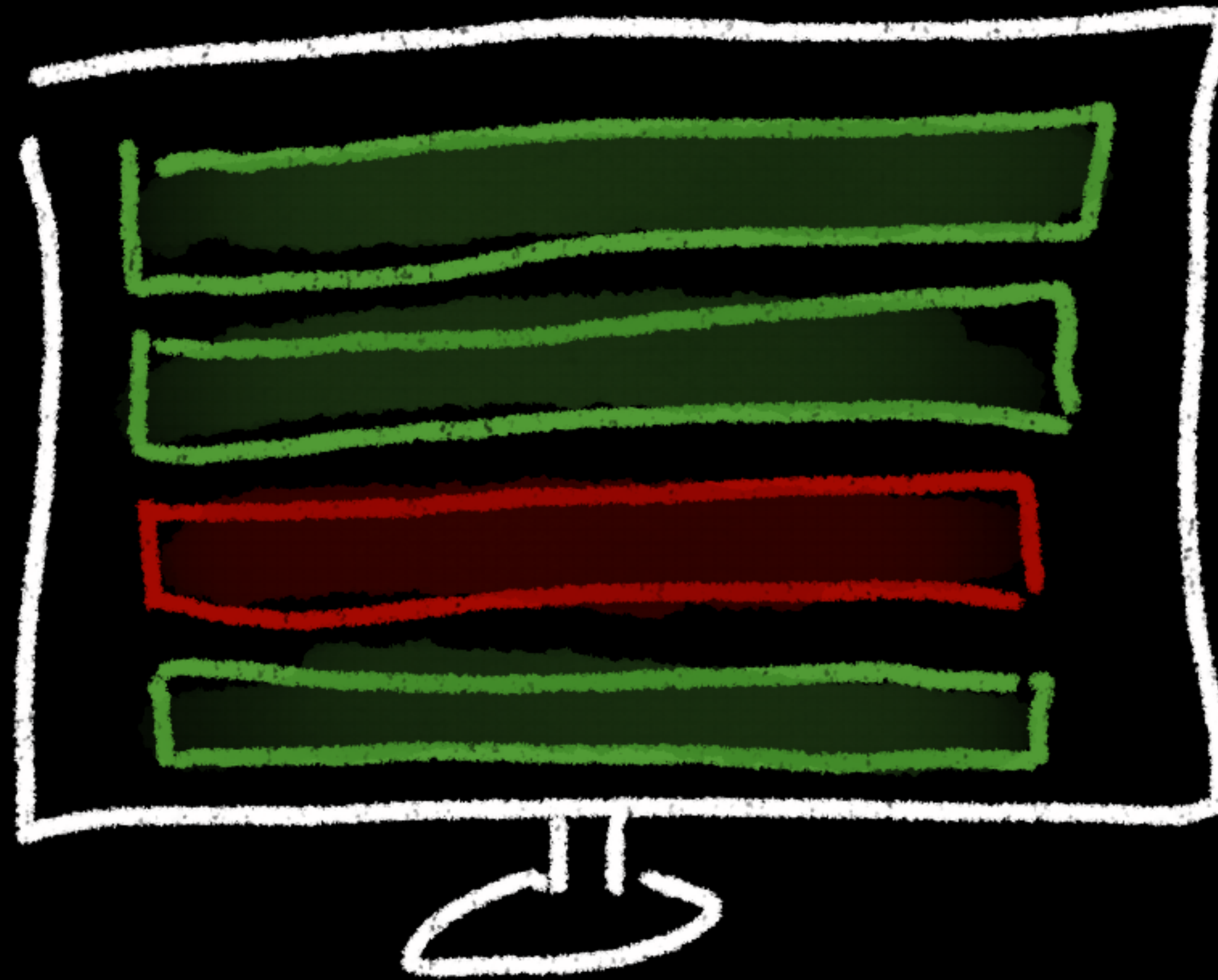


not a good CI/CD indicator

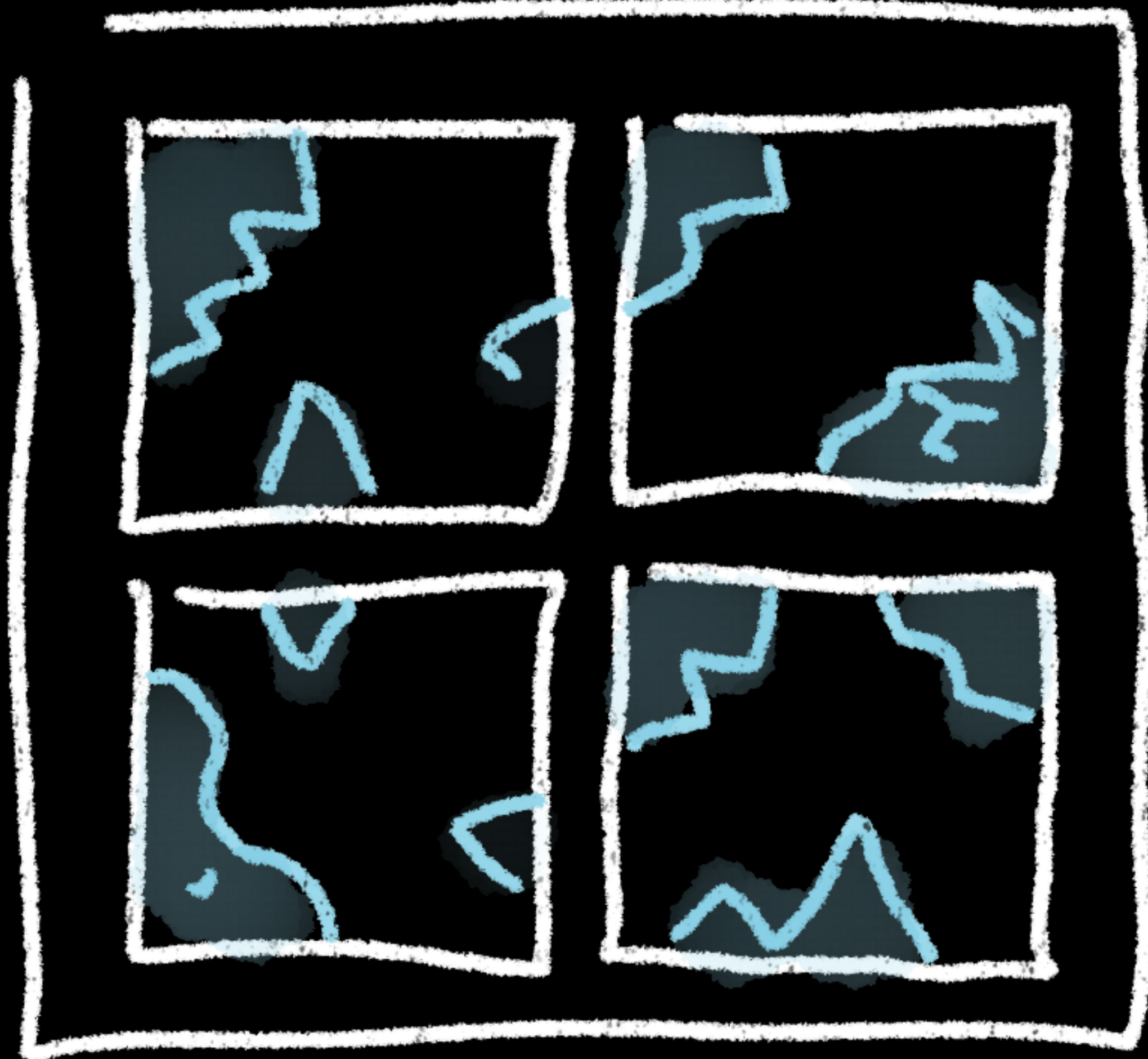


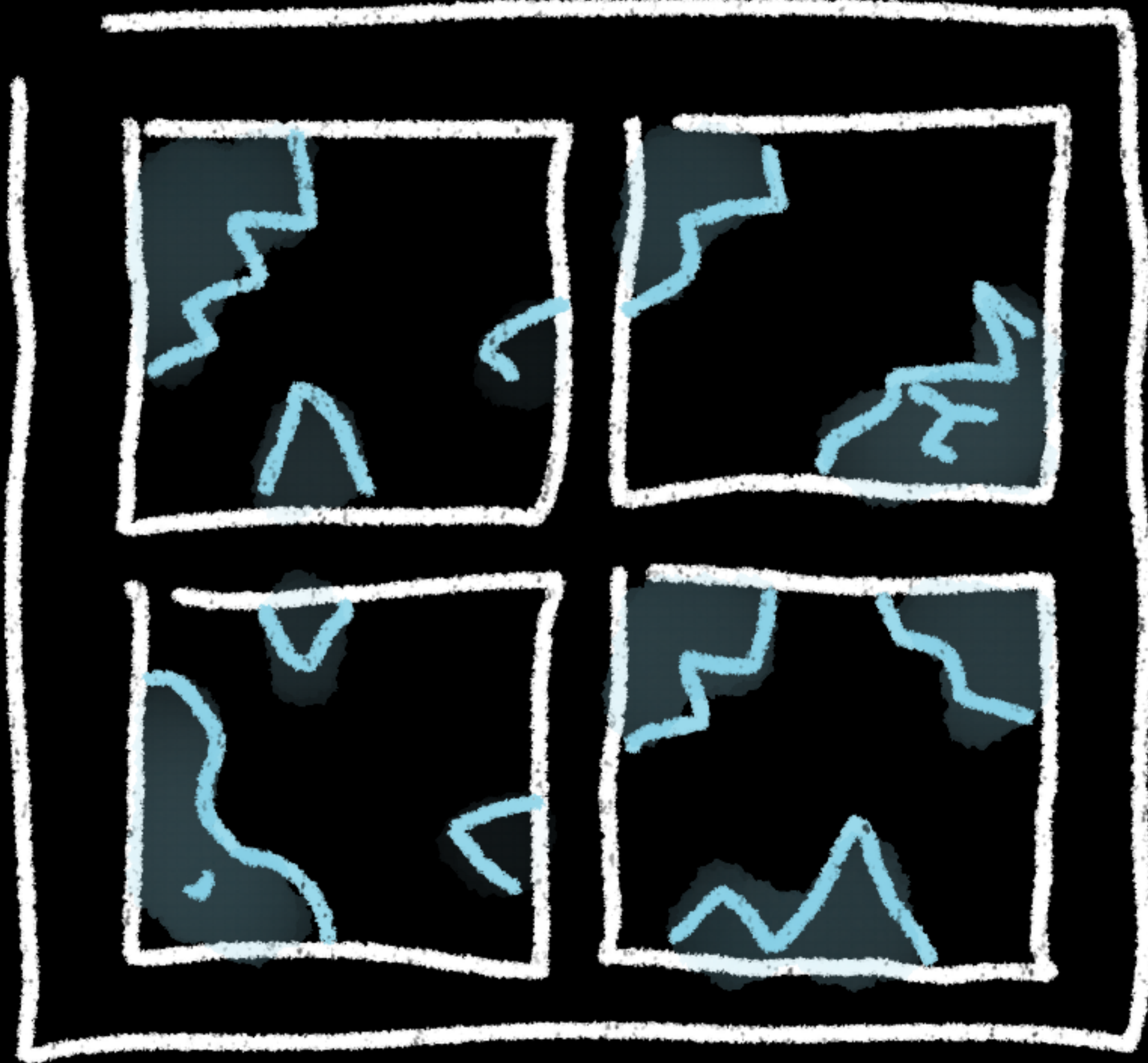
a good CI/CD indicator

“we don’t know when
the build is broken”



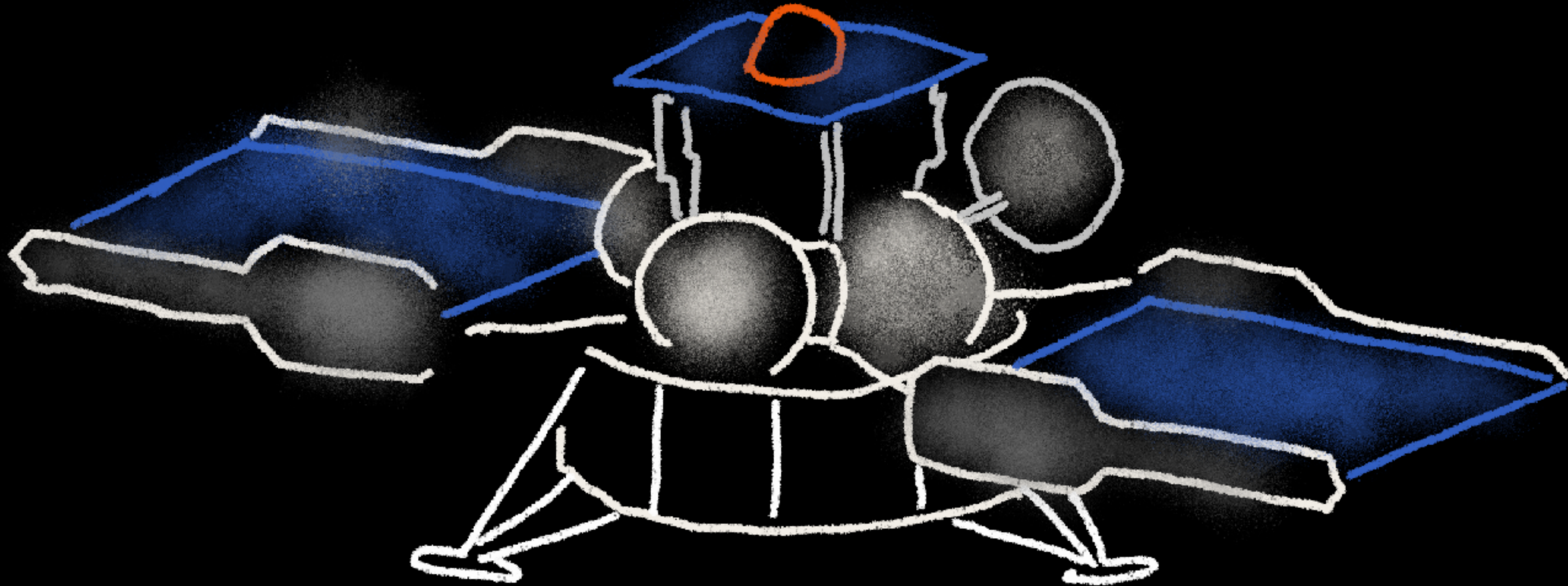
a good build radiator



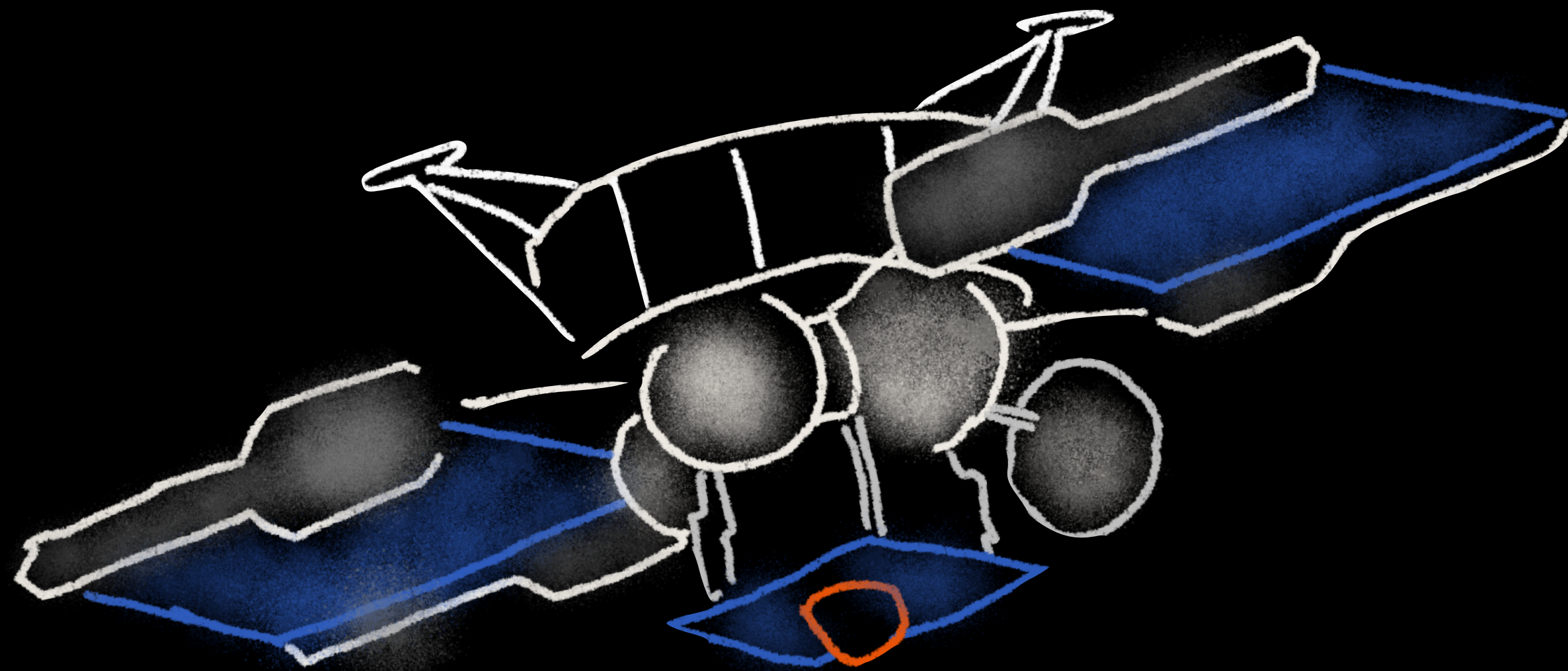


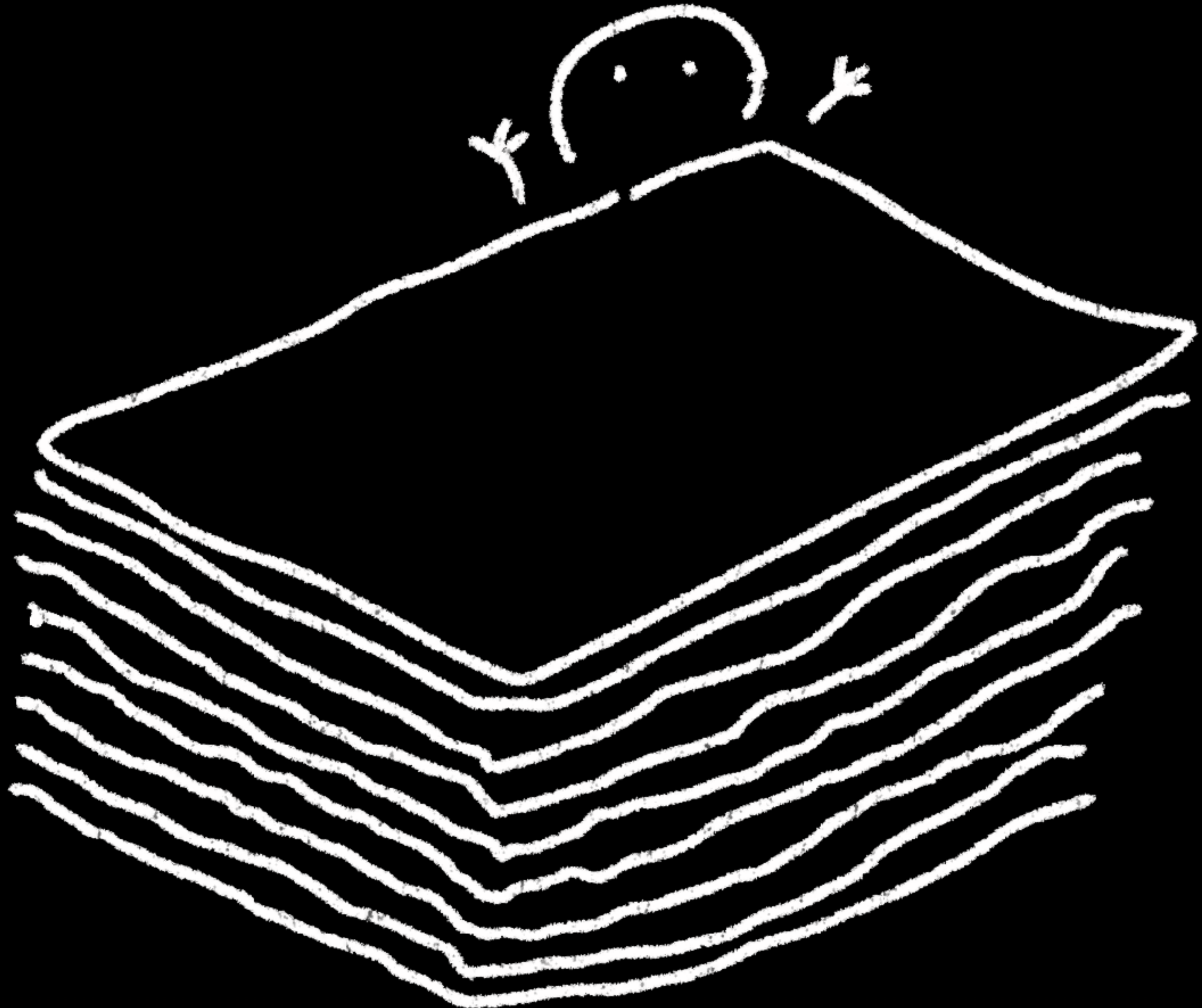
“oh yes, that build has been broken for a few weeks...”

how to brick a spaceprobe

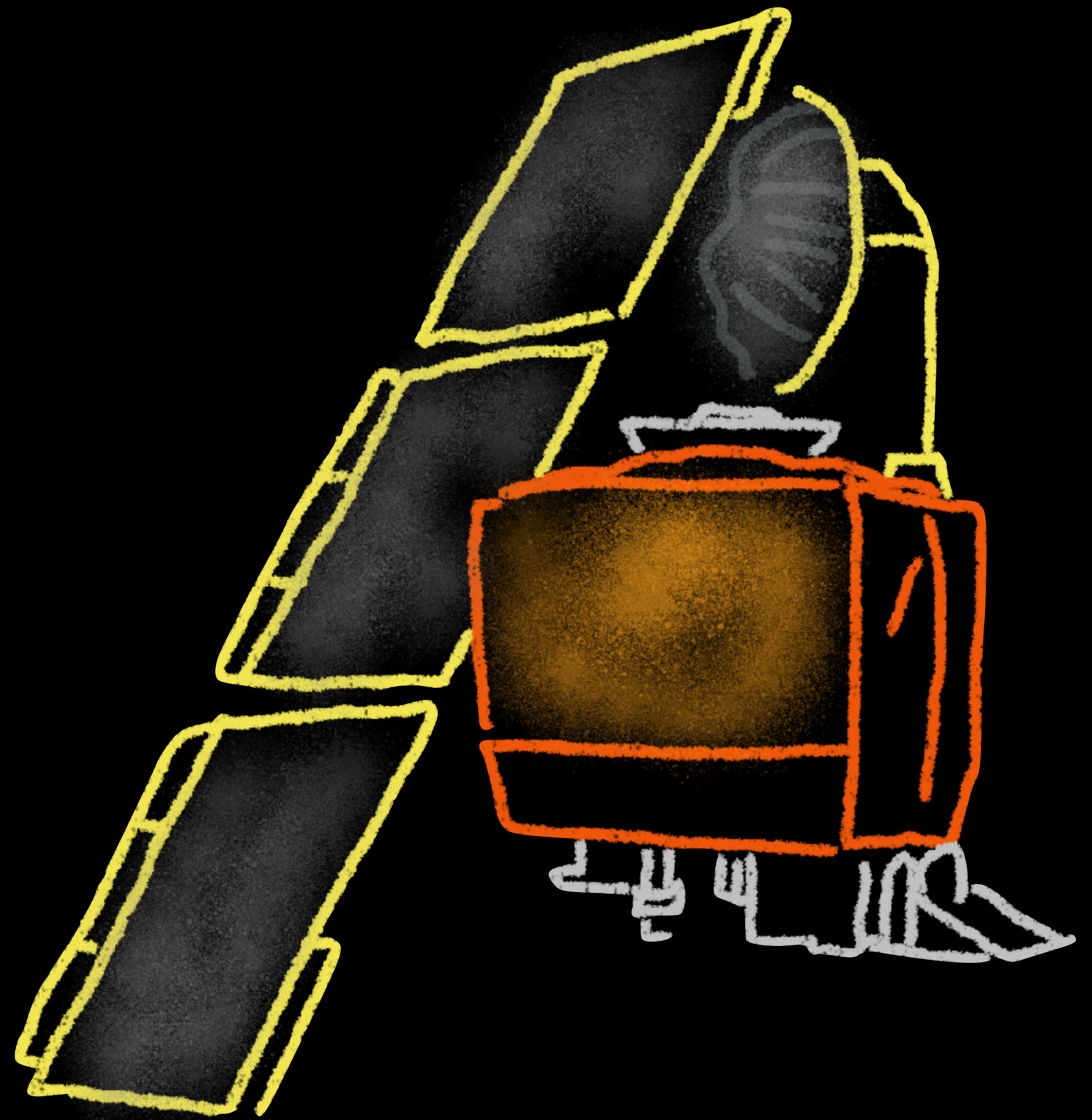


“we couldn’t get the automated checks to work, so we bypassed them”

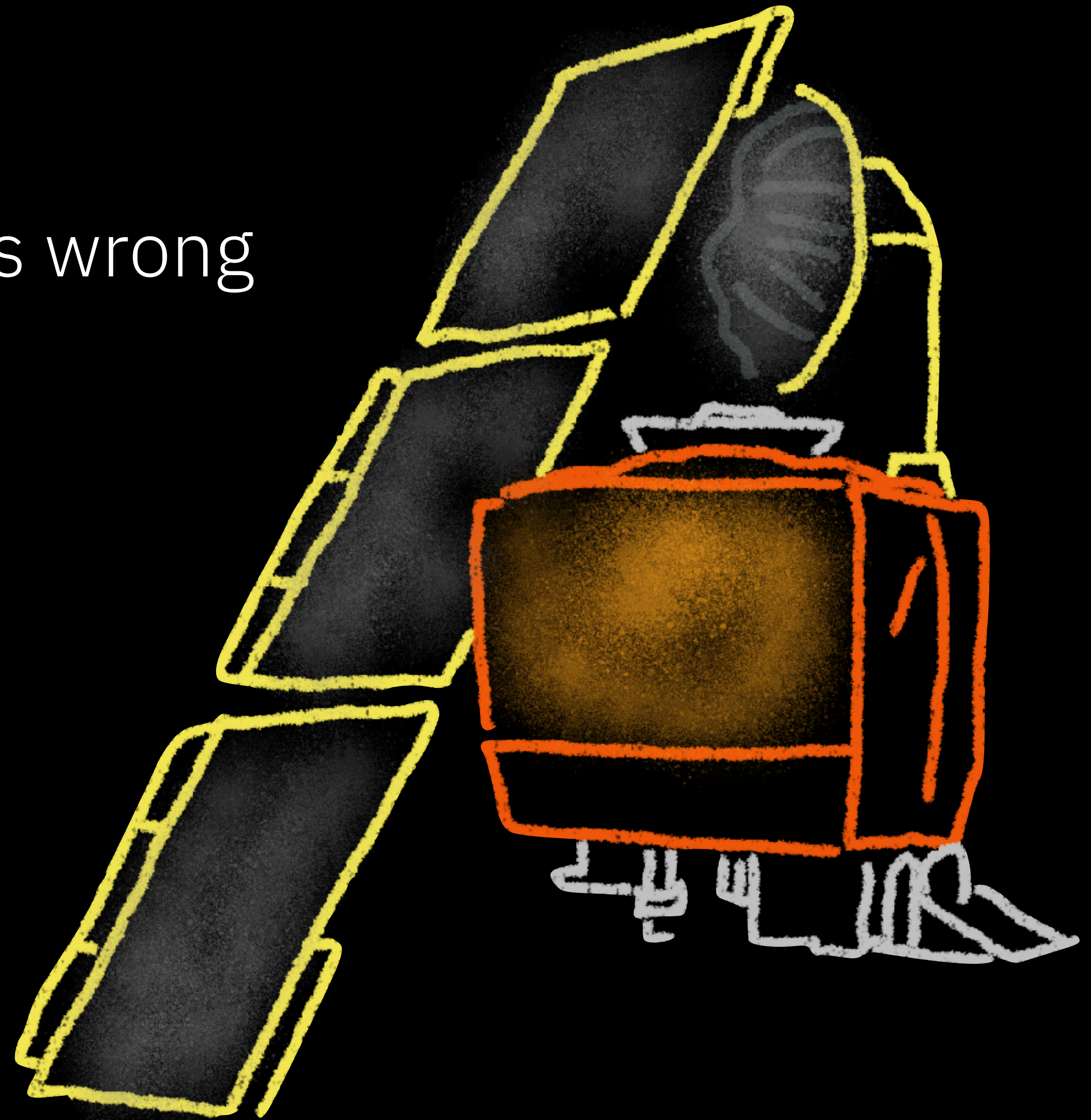




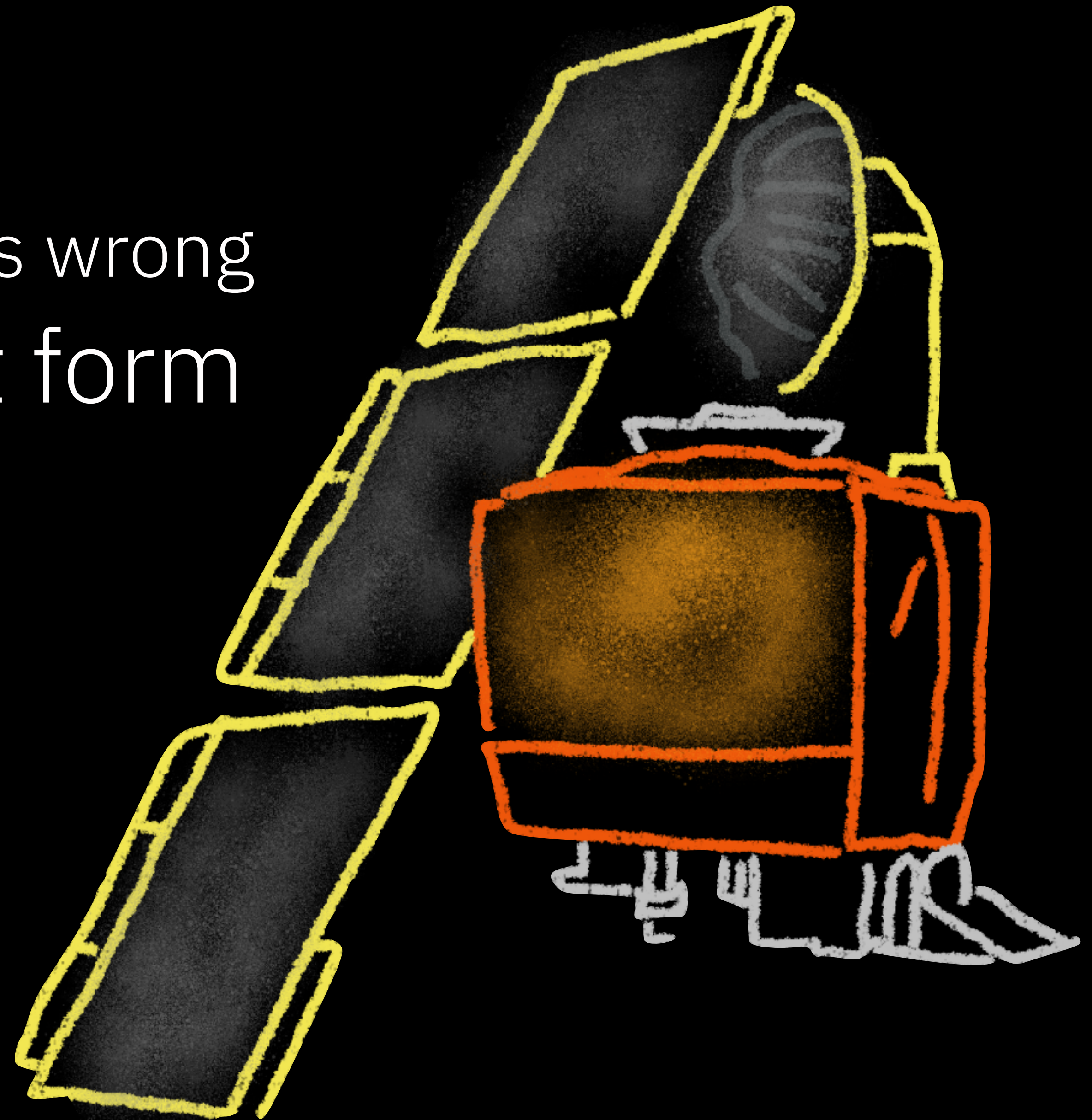
“we’ve
scheduled the
architecture
board review for
a month after the
project ships”



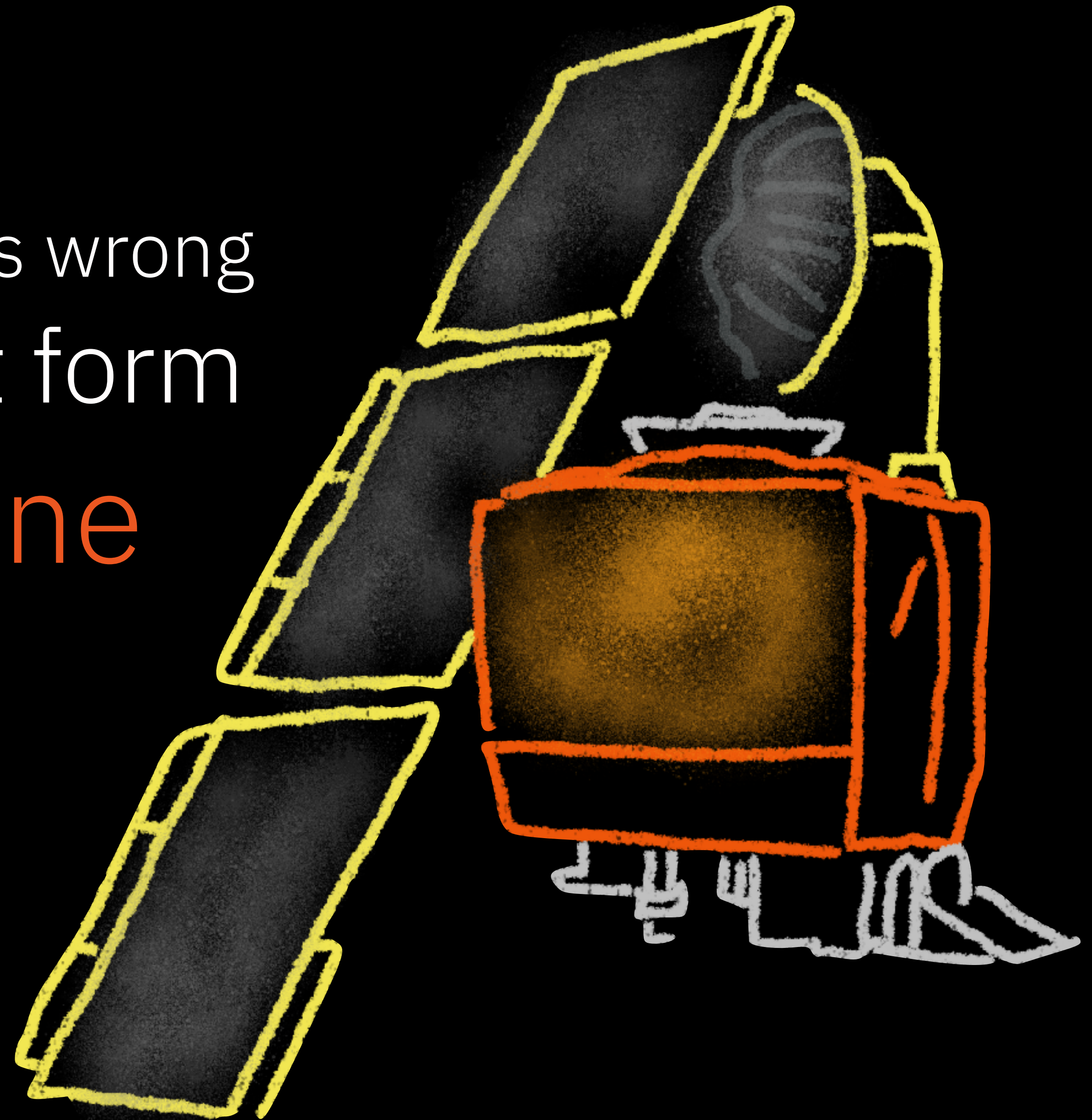
navigators warned something was wrong



navigators warned something was wrong
they didn't fill in the right form



navigators warned something was wrong
they didn't fill in the right form
so nothing was done



does the process add
value?

extreme programming is
the right kind of rigour

extreme programming is
the right kind of rigour

test-driven development

extreme programming is
the right kind of rigour

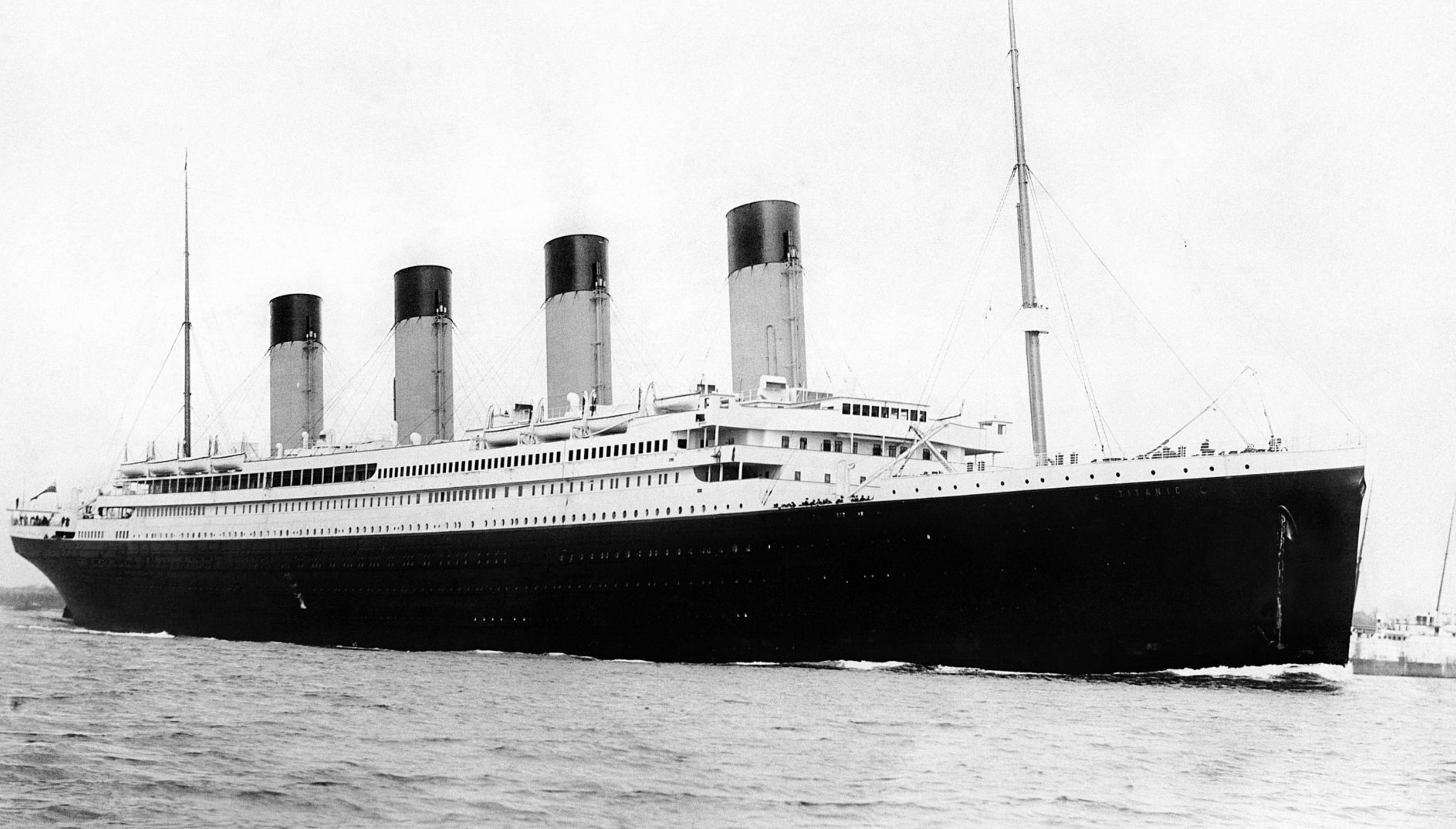
test-driven development
pair programming

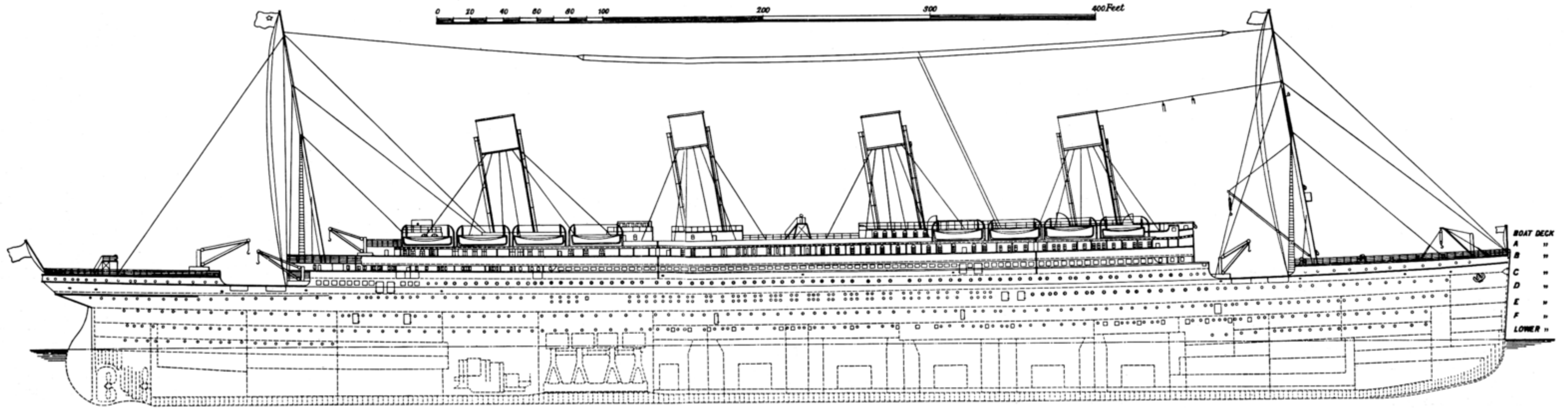
extreme programming is
the right kind of rigour

test-driven development
pair programming
optimise for feedback

“but it’s in the plan”

“but it’s not in the plan”





lots of bulkheads

it was too big

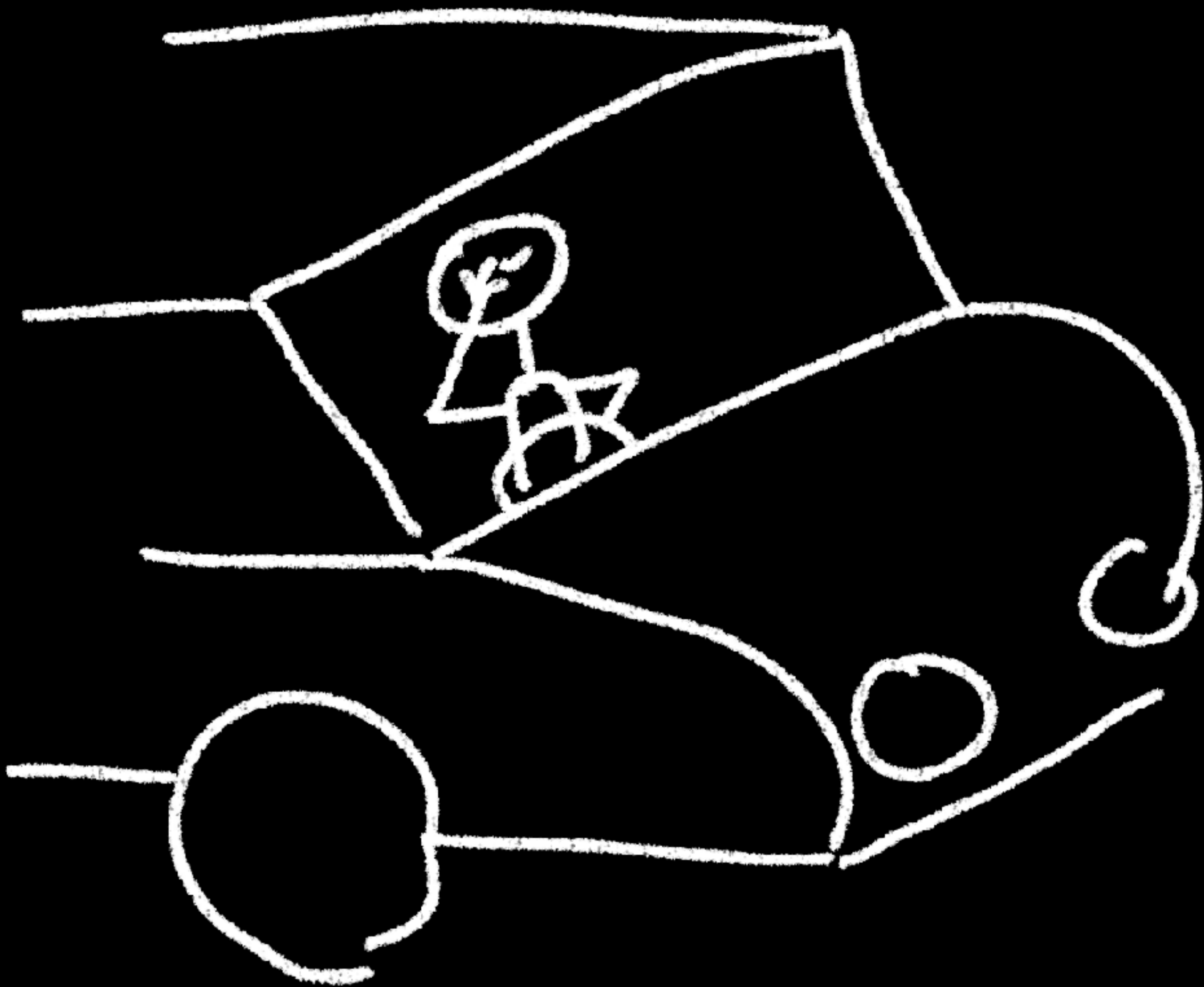
lookouts **saw** the iceberg
but the ship wasn't nimble enough to avoid it

Hidden Figures

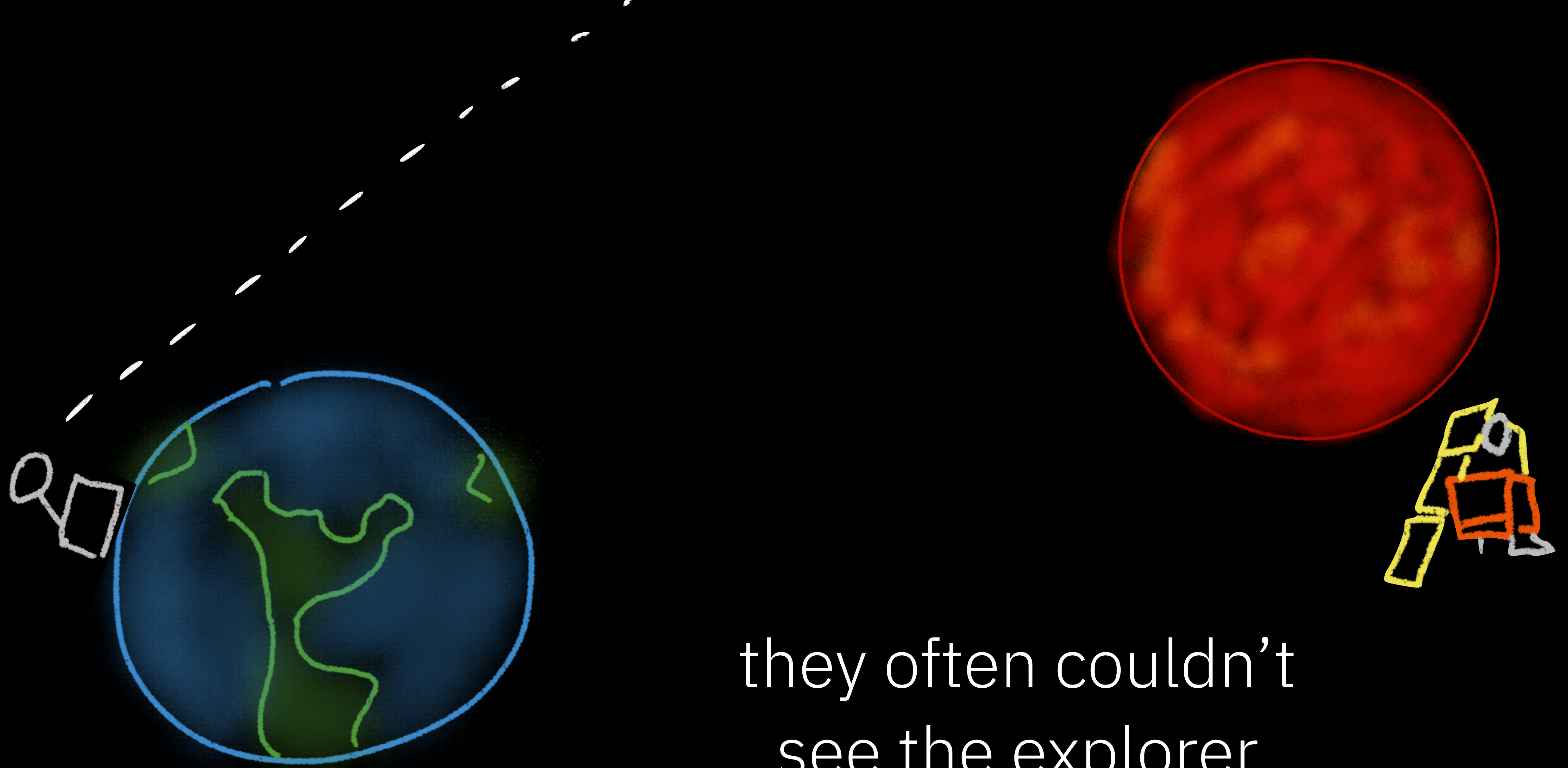




“we can’t ship until
every feature is
complete”



how **not** to
drive a car



they often couldn't
see the explorer

feedback is good business

feedback is good engineering

an mvp hurts

if you're not embarrassed
by your first release it
was too late

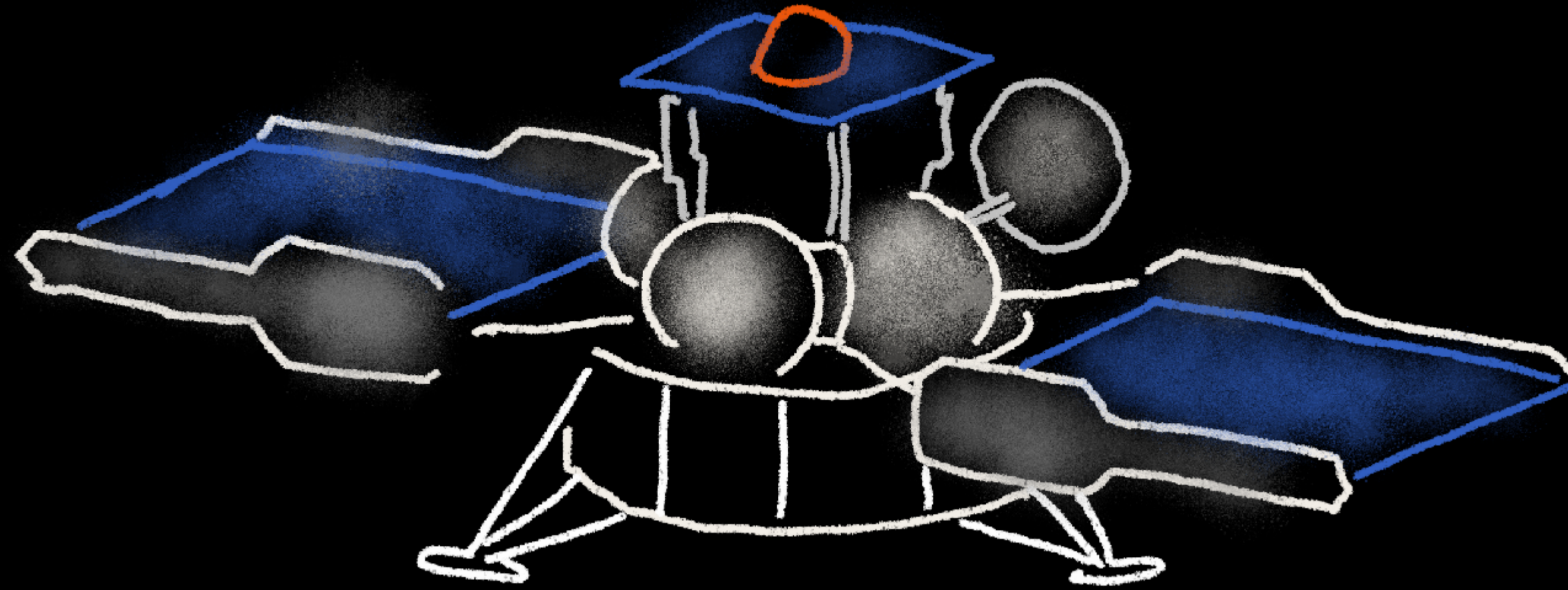
experiments can mean failure

experiments mean failure



users will have
weird behaviours

optimise for recovery

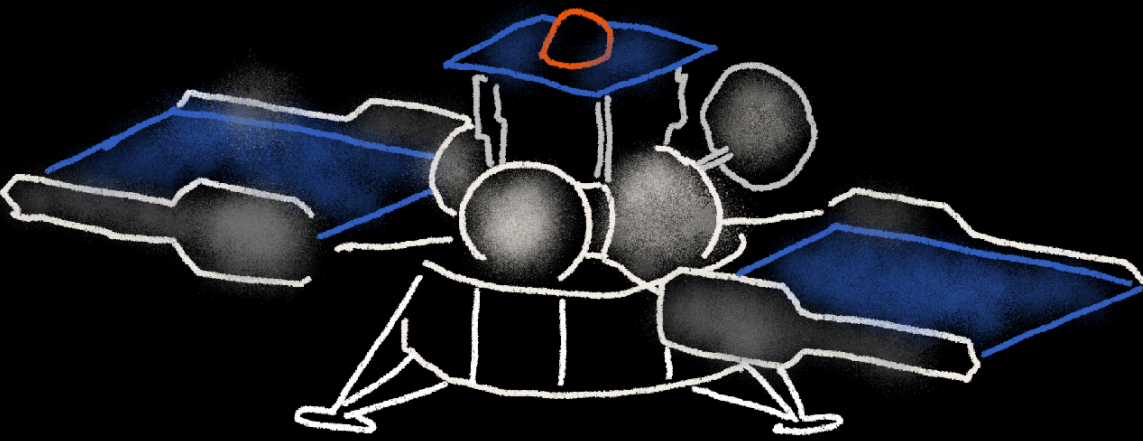


unrecoverable

back in ms
no data loss

manual
intervention

fast, but
data lost



bricked

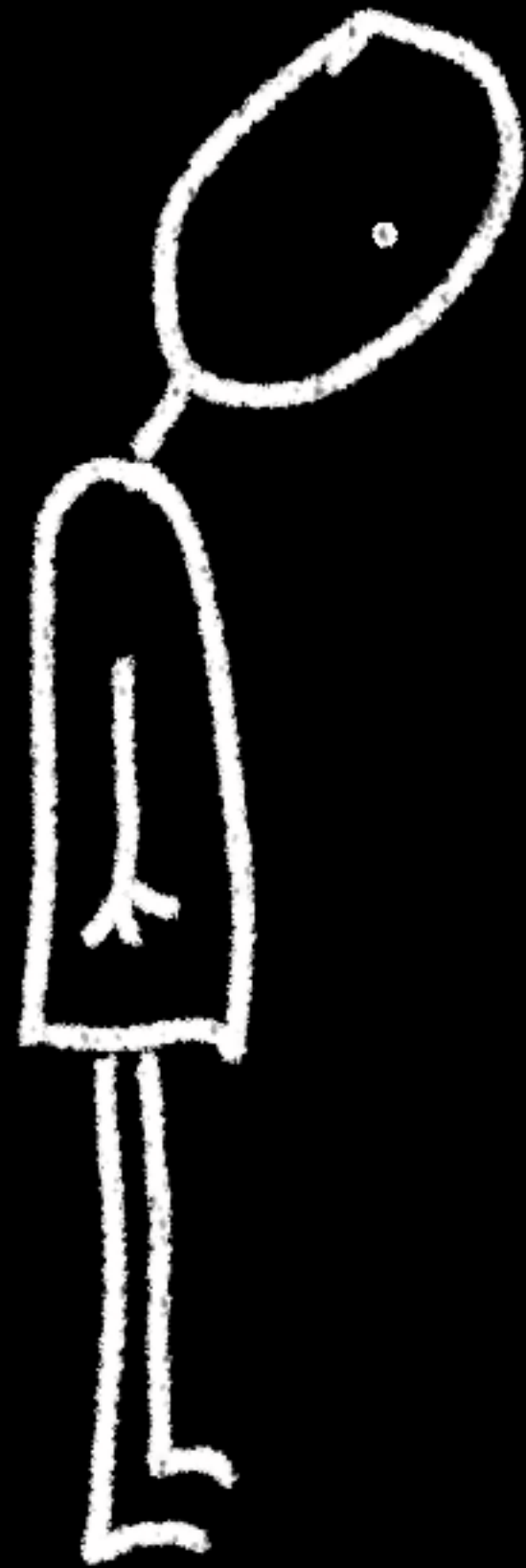
handoffs

business recoverability

remember, users will
have **weird** behaviours



speed



slow is
demoralising
for teams

fast is
good
business



A late change in requirements
is a competitive advantage.

–Mary Poppendieck



more feedback → more accuracy



cloud rescued
developers from tedium

cloud native
should feel
fun



thank you

@holly_cummins

#IBMCloudGarage